TRABAJO DE DIPLOMA

PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS.

Software de Propagación en Interiores para Sistemas RFID:

Análisis y Diseño

Autor:

Rolando Corratge Nieves

Tutores:

Ing.OmarA. GuzmánObregón

Ing. Serguei Guerra Fernández

Las producciones intelectuales serán el sustento fundamental de Cuba. La idea es convertir la informática en una de las ramas más productivas y aportadoras de recursos para la nación...

Fidel Castro

Declaración de Autoría

Por este medio declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas para que haga el uso del mismo en su beneficio. Para que así conste firmo la presente a los ____ días del mes de _____ del año 2011.

_____

Firma del Autor

Rolando Corratge Nieves

_____

Firma del Tutor

Omar A. Guzmán Obregón

_____

Firma del Tutor

Serguei Guerra Fernández

Dedicatoria

Agradecimientos

Contenido

Índice

Resumen

Los especialistas en el campo de las telecomunicaciones despliegan diferentes sistemas inalámbricos y uno de los requerimientos es que se le brinde cobertura adecuadamente al escenario de despliegue. Para ello antes de desplegar el sistema inalámbrico se debe realizar un estudio del escenario para calcular los dispositivos necesarios. Entre los diferentes sistemas inalámbricos existentes se encuentra la tecnología RFID la cual posibilita controlar y monitorizar diferentes objetos, en el siguiente trabajo se aborda las características, arquitectura y usos de dicha tecnología. Cuando se instala un sistema inalámbrico existen dos tipos de escenarios, los escenarios exteriores donde se trata de una ciudad o varias y el transmisor se encuentra a una altura superior a los edificios más altos de esta y los escenarios interiores donde se trata de brindar cobertura a un edificio, túnel o un campo conformado por varios edificios y el transmisor se encuentra dentro de estos. Existen diferentes métodos computacionales para realizar la simulación de sistemas inalámbricos y estos métodos son implementados en software que permiten el ahorro de recursos, mediciones y tiempo brindando los resultados previos antes de enfrentarse físicamente al escenario. El siguiente trabajo realiza un estudio previo de la tecnología RFID y su comportamiento en escenarios interiores, para luego brindar el análisis y diseño de una solución de un software capaz de realizar la simulación del radioenlace RFID en escenarios interiores. El análisis y diseño resultante de este trabajo constituyen la antesala de lo que será un sistema automatizado para realizar simulaciones y planificar el sistema RFID para que los especialistas en telecomunicaciones obtengan mejores resultados al desplegar dicho sistema.

Palabras Claves: Radio-propagación en interiores, RFID, Telecomunicaciones, simulación

Introducción

En la actualidad las telecomunicaciones se encuentran en su mejor desarrollo, nuevos sistemas de comunicaciones aparecen con alternativas para mejorar la funcionalidad de los sistemas y la prestación de servicios. A medida que han ido evolucionando las Tics[1] se ha profundizado en el estudio de la cobertura para diferentes sistemas de radio propagación. Con el aumento del uso de los sistemas inalámbricos se ha ido demandando mayores prestaciones y coberturas. Hace algunos años no era un problema pensar en la cobertura dentro de un ascensor o dentro de un medio de transporte. Pero hoy en día es necesario tener en cuenta diversos parámetros para planificar una red y darle cobertura a un interior.

La tecnología de Identificación por Radio Frecuencias RFID acrónimo del Inglés (Radio Frecuency Identification) es una tecnología de identificación por radiofrecuencias, que permite almacenar y enviar información de un objeto, animal o hasta de una persona. Se basa en la transmisión de datos por campos electromagnéticos y una identificación sin contacto visual directo, lo que permite que los procesos se agilicen de forma considerable, a diferencia de los sistemas de códigos de barras. La tecnología de RFID es utilizada en sistemas que tienen la habilidad de transmitir una identidad única, utilizando ondas de radio. La tecnología de identificación por radio frecuencia y su aplicación es relativamente nueva, se ha orientado al sector de almacenamiento y distribución de información teniendo además otros usos potenciales (1).

Al comienzo de la planificación de un proyecto de red, una de las cuestiones más críticas es la identificación de las limitaciones del proyecto, teniendo en cuenta los diversos problemas que pueden ocurrir potencialmente durante la ejecución y la implementación de este. Las antenas emisoras emiten una señal que experimenta múltiples cambios a lo largo de su trayectoria por el medio de propagación, llegando al receptor solo una pequeña parte. El camino entre receptor y emisor puede variar en múltiples formas debido a la existencia de diferentes obstáculos, complicándose así la predicción de la señal recibida en un determinado punto y el análisis del canal de radio. Para resolver este problema surgen los modelos de propagación que se enfocan en predecir la potencia promedio de la señal recibida, así como la variación de la potencia en la proximidad espacial de un lugar en particular. Estos modelos son difíciles de calcular manualmente por lo que en la actualidad son reflejados en un software que sea capaz de calcularlos.

En el siguiente trabajo proponemos el análisis y diseño de un software capaz de realizar la planificación de redes en interiores para los sistemas RFID. El software cuenta con un módulo donde se editará el escenario al cual se le desea dar cobertura dándole las propiedades a los materiales que componen los diferentes elementos de este (paredes, ventanas, etc.) y un módulo de propagación donde serán incorporadas los dispositivos con sus antenas en el escenario para simular el fenómeno de

[1] Tics: Tecnologías de la informática y las telecomunicaciones

propagación, determinar un aproximado de la estructura y los elementos necesarios para implantar RFID en este escenario.

Situación problémica: No existe una herramienta o solución disponible para los ingenieros de telecomunicaciones de Cuba costeable para simular el radio-enlace RFID, las aplicaciones de propagación en interiores existentes no realizan el análisis el de propagación para RFID o bien son propietarias. La simulación del radio-enlace RFID permite a los telecomunicadores ahorrar equipamiento que no siempre están disponibles para realizar las mediciones y permite a los alumnos de dicha disciplina que conocer mediante la simulación un aproximado de lo que ocurre en la realidad. En nuestra Universidad se está conformando además un grupo multidisciplinario de implementación (de hardware y software) que son los primeros interesados en que este proyecto se lleve a cabo. A partir de las razones explicadas se decide desarrollar el sistema.

Por lo que surge como **Problema a Resolver:** ¿Cómo contribuir a mejorar la observación y análisis de la propagación en interiores en sistemas RFID?

Para dar solución al problema planteado anteriormente se propone como **Objeto de estudio:** La propagación en interiores para sistemas RFID.

Teniendo como **Campo de acción:** La propagación en interiores.

La idea que se persigue como **Objetivo General:** Proponer el análisis y diseño de una herramienta que viabilice el análisis de la propagación en interiores en sistemas RFID.

Para desarrollar satisfactoriamente la investigación se han trazado las siguientes **Tareas de la investigación:**

- Revisión de los paquetes de SOFTWARE dedicados a la planificación de sistemas inalámbricos, concretamente los que abordan la propagación en interiores.
- Caracterización del canal radio y estudio de los fenómenos de la propagación en ambientes interiores.
- Descripción de los principales modelos de radio propagación en interiores.
- Acercamiento a los sistemas RFID, definición, características, y principales elementos que componen su arquitectura
- Identificación de las antenas RFID utilizadas en ambientes interiores. Parámetros fundamentales que inciden en la radio-propagación.
- Acercamiento a las herramientas CAD que se emplean en el diseño y modelación de ambientes interiores para la planificación de radioenlaces.
- Documentación y justificación de las herramientas para el desarrollo de la arquitectura de SOFTWARE.
- Definir los requisitos funcionales y parámetros de los principales sub módulos del SOFTWARE (Editor de Escenario y Propagación), así como la de los sub módulos hijos que surjan en el transcurso de la investigación.

- Validar y fundamentar la propuesta final del módulo de propagación en interiores para sistemas RFID, de modo que facilite el análisis por parte de los especialistas en Telecomunicaciones.

En el transcurso de la investigación se utilizarán Métodos de Investigación Científica, específicamente los Métodos:

- **Métodos teóricos:**
  - Revisión documental: Consiste en la selección y recopilación de información por medio de la lectura y crítica de documentos y materiales bibliográficos. Utilizada para recoger los datos más valiosos que aporta la bibliografía actual sobre el tema a tratar.
  - Analítico-Sintético: Permite el tránsito en el estudio de un fenómeno, del todo a las partes que lo componen y de estas al fenómeno pensado.
    En la parte analítica este método implica el análisis, esto es la separación de un todo en sus partes, o en sus elementos constitutivos. Se apoya en que para conocer un fenómeno es necesario descomponerlo en sus partes. Dentro de su síntesis implica una unión de elementos para formar un todo. Tendiendo a reconstruir su todo a partir de los elementos estudiados por el análisis. Es empleado en la recopilación de conceptos necesarios a dominar sobre los sistemas RFID y sobre la propagación de las ondas de radio, y en el análisis de las plataformas, metodologías y herramientas a utilizar en el desarrollo de la aplicación.
- **Métodos empíricos:**
  - Entrevista: Se define como un método de investigación mediante el cual se obtienen información amplia, abierta y directa de forma oral durante una conversación planificada entre el entrevistador y el o los entrevistados. Realizada de forma individual al los especialistas en telecomunicaciones con el objetivo de recopilar la mayor información vinculada al funcionamiento del sistema.

Estructura de la Tesis

El presente trabajo de diploma ha sido organizado en tres capítulos.

Capítulo 1: Contiene todo el fundamento teórico de la investigación. El mismo abarcará los principales conceptos y términos a tratar a lo largo del documento. Se hace referencia a la fundamentación teórica del tema en el cual se explica la necesidad del presente trabajo, así como las tecnologías, herramientas y lenguajes de programación candidatos.

Capítulo 2: En este capítulo se profundiza en el problema a resolver a través de su descripción y se muestra el modelo de dominio generado. Se identifican los requisitos funcionales y no funcionales a tener en cuenta. Por último, se hace una descripción de los Casos de Uso obtenidos a partir de los requisitos y se presentan los diagramas relacionados con estos.

Capítulo 3: En este capítulo se desarrolla el diagrama de clases del análisis y del diseño así como sus diagramas de interacción que describen como estas clase le dan respuesta a los casos de uso descritos en el capitulo anterior.

.

Capítulo 1. Fundamentación Teórica

Este capítulo contendrá todo el fundamento teórico de la investigación. El mismo abarcará los principales conceptos y principios teóricos de telecomunicaciones necesarios para poder realizar el análisis y diseño del software, los cuales estarán avalados por referencias actualizadas. En su desarrollo se explicará el funcionamiento de los sistemas RFID, los sistemas de antenas que este utiliza, la descripción del canal de radio, los diferentes modelos de propagación y los principales parámetros de las antenas.

1.1 Canal de Radio

El canal de radio es el canal de comunicación mediante el cual viajan las ondas electromagnéticas entre el receptor y el trasmisor estableciéndose el proceso de comunicación. El sistema de comunicación inalámbrico está constituido por un transmisor, que es el encargado de procesar la información y posteriormente radiarla. El receptor es el encargado de captar la señal, procesarla y así poder recuperar la información. Dentro de este canal existen dos formas de establecer la comunicación entre transmisor y receptor. El primero de ellos es cuando entre Tx[2] y Rx[3] por lo menos recibe una señal en forma directa, a este proceso se le llama enlace de Propagación en espacio libre. Otro, es cuando la señal transmitida entre Tx y Rx llega de manera indirecta, al cual se le conoce como enlace con Propagación en Línea con Obstáculos .

En el entorno exterior una de las entidades de comunicación normalmente se encuentra en una posición fija por encima de las construcciones de altura. El escenario interior, en la comunicación entre las entidades ocurre la dispersión y difracción de la señal causada por los objetos del entorno. El entorno de interiores se caracteriza por un gran número de rutas de propagación del transmisor al receptor, y por lo tanto es clasificada como un entorno multi-trayecto (6).

1.1.1 Propagación en espacio libre

Se denomina propagación en espacio libre a la propagación cuando no existen obstáculos en el camino entre el transmisor y el receptor. La perdida de señal que se produce está en función principalmente de la distancia que les separa, interviniendo igualmente otros factores como el tipo y diseño de las antenas, su patrón de radiación, etc. (6).

1.1.2 Propagación en Línea con Obstáculos

[2] Tx: Transmisor

[3] Rx: Receptor

En un entorno cerrado existirán multitud de obstáculos que se interpondrán en el trayecto seguido por las ondas electromagnéticas. Cada uno de estos obstáculos (paredes, suelos, muebles,…) afecta a la señal de determinada forma:

- Los objetos metálicos reflejan las señales de radio. Esto significa que la señal no atravesará muros metálicos y que los objetos metálicos dentro de una habitación reflejarán la señal y causarán desvanecimientos y atenuaciones de la señal.

- Madera, cristal, plástico y ladrillo reflejan parte de la señal de radio y dejan pasar parte del resto.

- Los objetos que contienen un alto grado de humedad absorben la mayor parte de la señal(6).

1.1.3 Reflexión y refracción

La reflexión y la refracción ocurren cuando una onda electromagnética que se propaga por el aire incide contra un objeto de grandes dimensiones en comparación con la longitud de onda de la señal. El resultado puede ser que la señal sea absorbida, reflejada o una combinación de ambas. Esta reacción depende principalmente de:

- Propiedades físicas del obstáculo, como pueden ser su geometría, textura y composición.

- Propiedades de la señal, como el ángulo de incidencia, orientación y longitud de onda.

Los conductores perfectos reflejarán la totalidad de la señal. Otros materiales reflejarán solo una parte de la energía incidente y transmitirán el resto. La cantidad exacta de refracción y reflexión depende igualmente del ángulo de incidencia así como del grosor y propiedades dieléctricas del material. Ver figura 1.1.

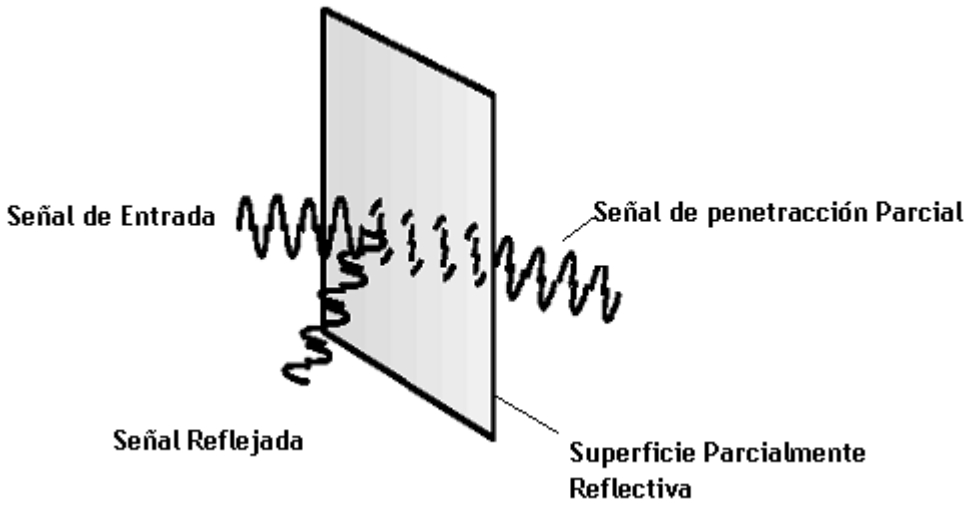


Figura 1.1 Fenómeno de reflexión y refracción

1.1.4 Desvanecimiento por multi-trayecto debido a reflexiones

Cuando una señal electromagnética es transmitida por el aire, es muy probable que debido al fenómeno de reflexión, alcance al receptor por múltiples caminos. Las señales procedentes de caminos alternativos llegarán ligeramente retardadas con respecto a la señal directa, lo que provoca el efecto de desvanecimiento. Esto es debido a que las señales reflejadas tendrán diferente amplitud que la señal directa cuando lleguen al punto de recepción retardadas.

1.1.5 Transmisión

La transmisión (o penetración) se produce cuando la señal se encuentra en su camino con un obstáculo que es, por así decirlo, transparente para las ondas de radio. Cuando la señal penetra un obstáculo experimentará una perdida, la cual será función del grosor del objeto y del material del que está compuesto. La frecuencia de la onda electromagnética también influye en qué proporción de la señal incidente atraviesa el objeto. Existen multitud de estudios sobre pérdidas estimadas en diferentes materiales.

1.1.6 Difracción

En óptica se entiende por difracción la desviación del rayo luminoso al rozar el borde de un cuerpo opaco. Las ondas difractadas se forman cuando el camino de propagación de la onda de radio es obstruido por una superficie que tiene irregularidades o bordes puntiagudos o angulados. La difracción

ocurre cuando los obstáculos son impenetrables por las ondas de radio. El resultado son ondas secundarias alrededor y detrás del obstáculo (figura 1.2).

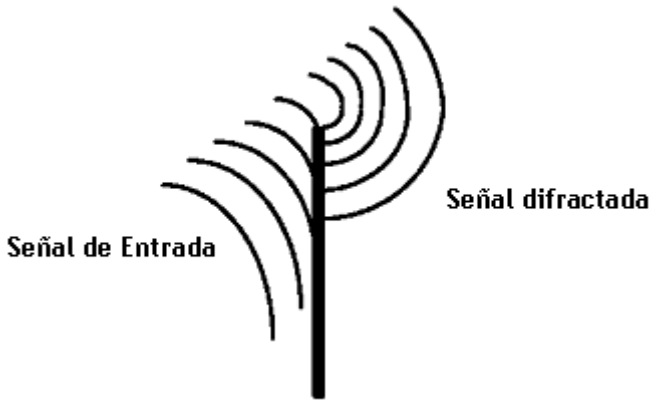


Figura1.2 Fenómeno de difracción

Los escenarios cerrados contienen muchos tipos de objetos con estas características orientados tanto en el plano vertical como horizontal. La señal difractada depende de la geometría del objeto así como la amplitud, fase y polarización de la onda incidente en el punto de difracción.

1.1.7 Dispersión

La dispersión ocurre cuando en el camino la señal se encuentra con objetos cuyas dimensiones son pequeñas en relación a la longitud de onda. El resultado es que el frente de onda se rompe o dispersa en múltiples direcciones (figura 1.3) (6).

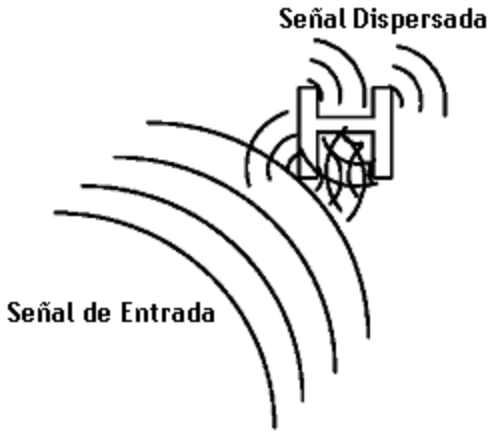


Figura 1.3 Fenómeno de Dispersión

1.2 Modelos de propagación en interiores

Un modelo de propagación es un conjunto de expresiones matemáticas, diagramas y algoritmos usados para representar las características de radio de un ambiente dado. Existen diferentes tipos de modelos, lo que permite que un fenómeno se represente en varios modelos, para esto es necesario conocer cuando un modelo es correcto, existen varios factores:

- ✓ El modelo puede explicar las observaciones realizadas del fenómeno.
- ✓ Se puede usar el modelo para predecir el comportamiento de fenómeno modelado.
- ✓ El modelo es consistente con otras ideas acerca del funcionamiento del fenómeno modelado.

Generalmente los modelos se pueden clasificar en empíricos o estadísticos, teóricos o deterministas o una combinación de estos dos (semi-empíricos) (6).

1.2.1 Modelos Empíricos

Se basan en la extrapolación estadística de resultados a partir de medidas del fenómeno realizadas sobre el terreno. La principal ventaja de este tipo de modelos es que se tienen en cuenta de forma implícita las influencias propias del entorno en su conjunto, sin ser reconocidas cada una de ellas de manera aislada.

Por el contrario, su precisión depende no sólo de la precisión de las medidas, si no de la similitud entre el entorno donde fueron llevadas a cabo las medidas y el entorno a analizar. La eficiencia de estos modelos suele ser satisfactoria computacionalmente.

Modelo One Slope

Es un modelo muy rápido, porque depende sólo en la distancia r entre el transmisor y el receptor(figura 1.4). El único parámetro que es considerado por lo tanto el exponente de pérdida de trayectoria.

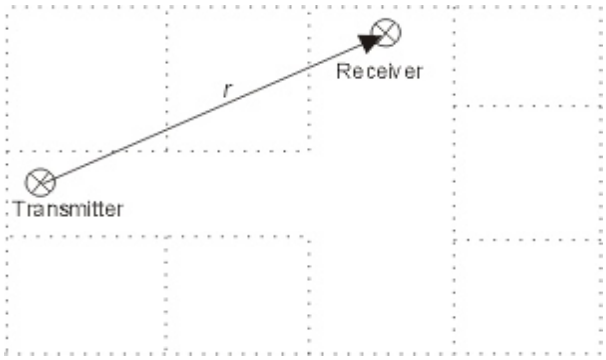


Figura 1.4 Modelo One Slope

Con el exponente de pérdida de trayectoria, el modelo es calibrado para cada escenario. Las paredes y otros elementos en la construcción de la base de datos no se consideran en el modelo de una pendiente.

Modelo Motley Keenan

Este modelo considera todas las paredes de intersección del rayo directo entre el transmisor y el receptor. Una atenuación puede ser definida por el usuario para las paredes Lw y los límites de los suelos Lc. Esta atenuación se utiliza para todas las paredes y los techos de intersección del rayo directo (figura 1.5).

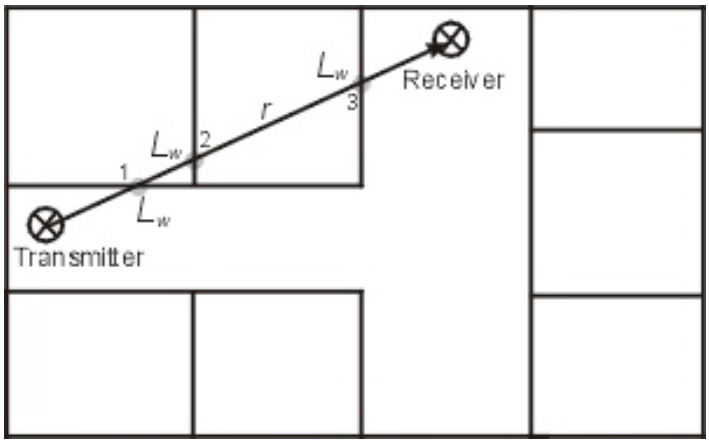


Figura 1.5 Modelo Motley Keenan

Modelo COST 231 Multi-Wall

El modelo COST 231 es un modelo empírico que tiene en cuenta las pérdidas de propagación en espacio abierto así como las perdidas introducidas por las paredes, suelos y techos penetrados en el trayecto directo entre transmisor y receptor (figura 1.6) (7).

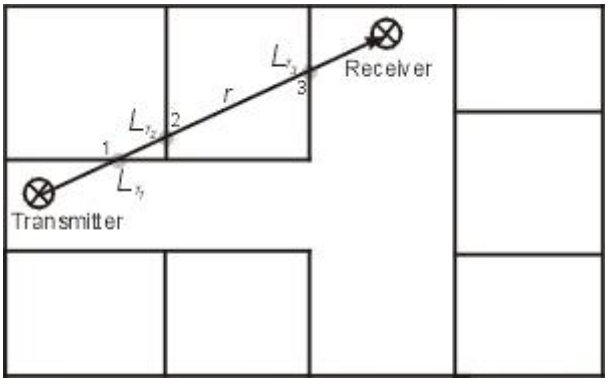


Figura 1.6 COST 231 Multi-Wall Model

El modelo se basa en el modelo COST 231 definido como:

$$L = L_{FS} + L_C + \quad K_{Wi}L_{Wi} + n^{\frac{n+2}{n+1}b} \quad * L_f \qquad (1)$$

Donde, L_{FS} son las pérdidas en espacio libre entre transmisor y receptor, L_C la constante de perdida (37dB), K_{Wi} el número de paredes de tipo i penetradas, L_{Wi} las pérdidas debida a muro de tipo i, n el número de suelos penetrados, b un parámetro empírico, y L_f las pérdidas entre suelos adyacentes(6).

1.2.2 Modelos Determinísticos

Se basan en principios fundamentales de la física en cuanto a propagación de ondas de radio y los fenómenos que la rodean. Pueden ser aplicados en diferentes entornos sin afectarles a su precisión. Los algoritmos usados por los modelos deterministas son generalmente muy complejos y computacionalmente poco eficientes. Actualmente son los más utilizados debido a que ofrecen mejores resultados que los empíricos en cuanto a precisión.

El Algoritmo de Trazado De Rayos o Ray Tracing

El algoritmo de trazado de rayos, calcula todos los posibles caminos desde el transmisor al receptor. En los modelos básicos de trazado de rayos, la predicción se basa en cálculos de transmisión en espacio abierto complementados con el efecto de reflexión de las paredes, teniendo en cuenta una única reflexión. Los algoritmos de trazado de rayos más complejos incluyen mecanismos de difracción, dispersión difusa y penetración a través de diversos materiales. En conclusión, el nivel de señal en un punto determinado se obtiene mediante la suma de los componentes de todos los caminos posibles existentes entre transmisor y receptor. Además de las pérdidas de propagación, en estos modelos se pueden predecir de manera fiable la dispersión en el tiempo de la señal. Hoy en día los modelos de trazado de rayos están considerados entre los más precisos de entre los modelos de predicción de potencia de campo; sin embargo requieren una representación muy detallada del área a analizar. La precisión del modelo depende en gran medida de la precisión y completitud/complejidad de la base de datos asociada a la representación. Por otro lado, el tiempo computacional depende en modo exponencial de estos detalles. Así, el tiempo de computación de un área reducida pero muy detallada puede ser mucho mayor que el de un área mayor pero con menor nivel de detalle (6).

1.3 Acercamiento a los sistemas RFID

La identificación por radiofrecuencia o RFID, como es comúnmente conocido, es un término genérico que se refiere a la información y varias tecnologías de comunicación que comparten la capacidad para identificar automáticamente objetos, lugares y personas y enviarlos a los sistemas de cómputo, sin necesidad de intervención manual.

La identificación automática con RFID se compone por: el objeto, lugar o persona que está marcado con un código único de identificación contenido en una etiqueta de RFID, que es de algún modo conectado o integrado en el objetivo y un dispositivo de computación que se necesita para identificar el destino que cuenta con un lector de RFID para la búsqueda de las etiquetas. Cuando se recibe una indicación de que una etiqueta está presente en sus proximidades, se instruye al lector a que solicite el código. Los datos recuperados se registran de modo que sea propicio para el uso de la aplicación específica.

RFID es una tecnología inalámbrica la identificación automática que utiliza una etiqueta integrada en la entidad de destino para marcarlo con un código único. Las etiquetas son leídas por un lector adecuado, un dispositivo que proporciona la energía para el funcionamiento de la etiqueta y para la comunicación. Hay múltiples variantes de RFID, pero una diferencia principal entre aquellos que utilizan el componente magnético y los que utilizan el componente eléctrico de la onda emitida por el lector. Cada alternativa tiene características particulares con ventajas únicas y limitaciones. La tecnología tiene una historia de más de 70 años desde el descubrimiento de su principio de operación, pero ha requerido de muchos avances tecnológicos para convertirse en práctica(8).

1.3.1 Rango de frecuencias

Los sistemas RFID se clasifican dependiendo del rango de frecuencias que usan. Existen cuatro tipos de sistemas: de frecuencia baja (entre 125 ó 134,2 kilohercios); de alta frecuencia (13,56 megahercios); UHF o de frecuencia ultra elevada (868 a 956 megahercios); y de microondas (2,45 gigahercios). Los sistemas UHF no pueden ser utilizados en todo el mundo porque no existen regulaciones globales para su uso(9).

1.3.2 Arquitectura

Un sistema RFID consta de los siguientes tres componentes básicos(10):

- Etiqueta o Tag RFID
- Lector o interrogador RFID
- Subsistema de procesamiento de datos o Middleware RFID

Etiqueta o TagRFID :

Está compuesta por una antena, un transductor radio y un material encapsulado o chip. El propósito de la antena es permitirle al chip, el cual contiene la información, transmitir la información de identificación de la etiqueta. Existen varios tipos de etiquetas. El chip posee una memoria interna con una capacidad que depende del modelo y varía de una decena a millares de bytes. Existen varios tipos de memoria:

- Solo lectura: el código de identificación que contiene es único y es personalizado durante la fabricación de la etiqueta.
- De lectura y escritura: la información de identificación puede ser modificada por el lector.
- Anticolisión. Se trata de etiquetas especiales que permiten que un lector identifique varias al mismo tiempo (habitualmente las etiquetas deben entrar una a una en la zona de cobertura del lector) (10).

Lector o interrogador RFID

Está compuesto por una antena, un transceptor y un decodificador. El lector envía periódicamente señales para ver si hay alguna etiqueta en sus inmediaciones. Cuando capta una señal de una etiqueta (la cual contiene la información de identificación de esta), extrae la información y se la pasa al subsistema de procesamiento de datos.

Para entender cómo una etiqueta RFID notifica al lector acerca de su presencia e identidad, tenga en cuenta el escenario simple muestra en la Figura 1.7. El lector RFID transmite señales de radio en una frecuencia predeterminada y en un intervalo (por lo general cientos de veces por segundo). Las etiquetas de radiofrecuencia que se encuentran en el rango de este lector tomará su transmisión porque cada una tiene una antena incorporada que es capaz de escuchar señales de radio a una frecuencia preestablecida Las etiquetas pueden usar la energía de la señal del lector para reflejar la señal de retorno. Las etiquetas pueden modular la señal a enviar la información de nuevo al lector, tales como un número de identificación (10).

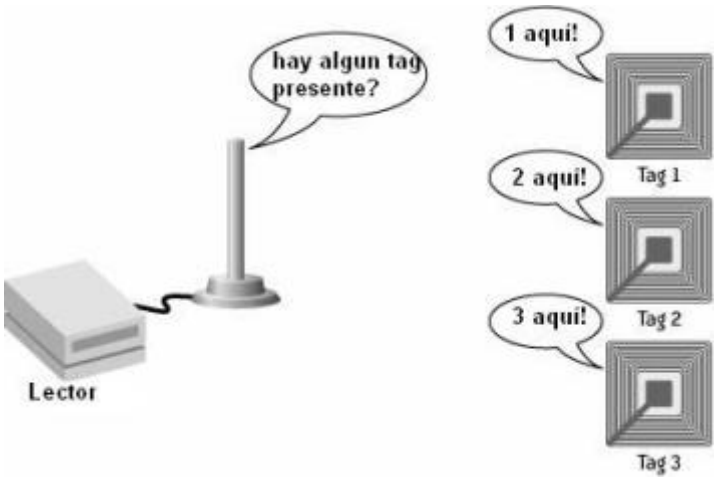


Figura 1.7Comunicación entre el Lector y las Etiquetas RFID

Subsistema de procesamiento de datos o Middleware RFID:

Software que proporciona los medios de proceso y almacenamiento de datos. Estandariza la forma de hacer frente a la avalancha de información que estas etiquetas pequeñas producen. Además del evento de filtrado, que también necesitan un mecanismo para encapsular las aplicaciones con el fin de evitar que conocer los detalles de la infraestructura física (lectores, sensores, y sus configuraciones) (10).

1.4 Herramientas y tecnologías

A continuación se describen las metodologías, herramientas y tecnologías utilizadas tanto para realizar el análisis y diseño del software y las que se proponen a utilizar para llevar a cabo su implementación.

1.4.1 Metodología de Desarrollo Utilizada

Rational Unified Process (RUP) Es la metodología más utilizada en la actualidad para proyectos a medio y largo plazo. El nivel de detalle con el que la metodología RUP propone trabajar ayuda a lograr el producto esperado. Divide en cuatro fases el desarrollo del software, en las cuales se realizan actividades que van dando lugar a los artefactos necesarios para el avance del proyecto. Tales actividades son conocidas como flujos de trabajo, los cuales tienen mayor o menor peso según la fase en que se encuentre el proyecto (figura 1.27). Existen nueve flujos, de ellos los seis primeros son conocidos como flujos de ingeniería y los otros tres como flujos de apoyo. A continuación se relacionan las fases y los flujos de trabajo.

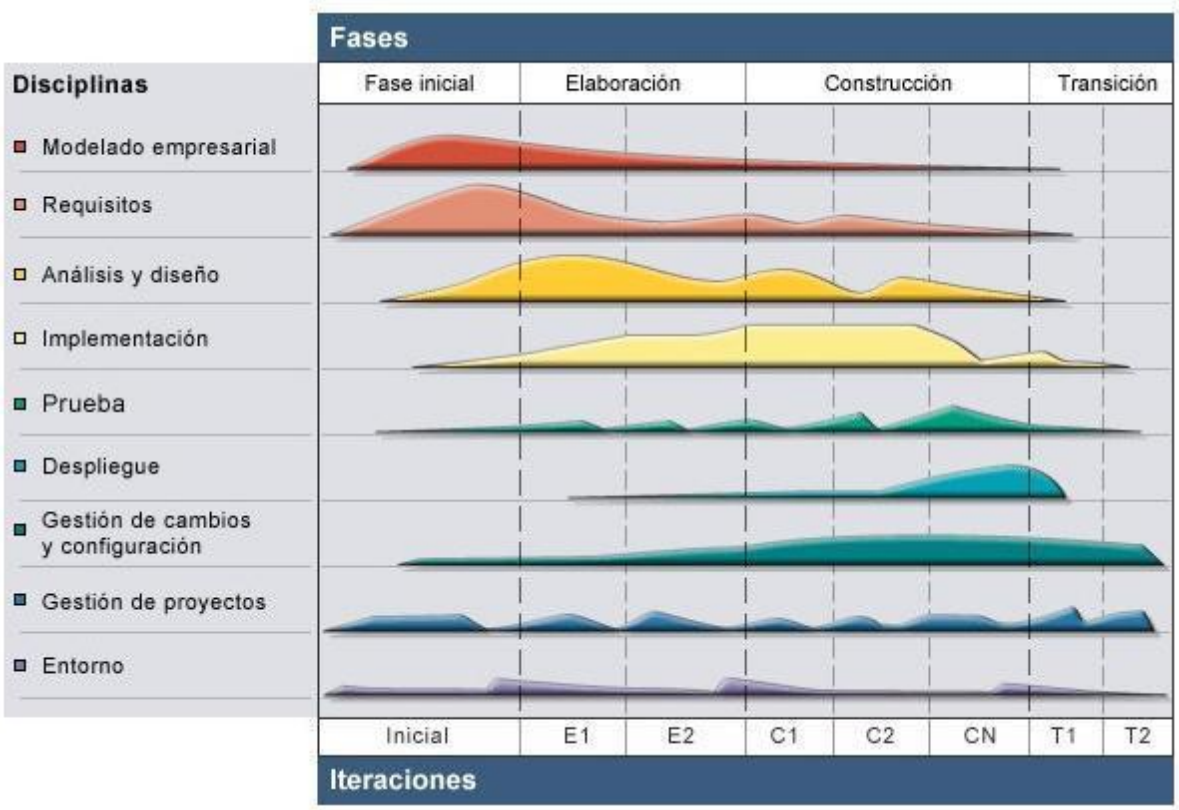


Figura 1.8RUP en dos dimensiones.

Fases de RUP:

- **Inicio:** En esta etapa se describe el negocio y se determinan los límites del proyecto.
- **Elaboración**: Se define la línea base de la arquitectura.
- **Construcción**: Se obtiene un producto documentado, listo para su uso, se obtienen uno o varios release que han pasado por pruebas.
- **Transición**: Se obtiene un producto final listo para su uso. Puede implicar reparación de errores.

Flujos de Trabajo

- **Modelación del Negocio**: Se describen y se detallan las necesidades del negocio.
- **Requerimientos**: Se traducen las necesidades del negocio a un sistema automatizado.
- **Análisis y Diseño**: Los requerimientos se trasladan dentro de la arquitectura de software.
- **Implementación**: Se comienza a crear el software ajustándose a la arquitectura para que tenga el comportamiento esperado.
- **Prueba:** Se verifica si el comportamiento logrado es correcto y si el desarrollo ha sido acorde a la arquitectura.
- **Administración de proyecto:** Se administran horarios y recursos.
- **Administración de Configuración y Cambios:** Se controlan las versiones del proyecto.
- **Ambiente:** Actividades que describen los procesos y herramientas que soportarán el equipo de trabajo del proyecto.

Características de RUP:

Dirigido por Casos de Uso: Los casos de uso reflejan lo que los usuarios futuros necesitan y desean, lo cual se capta cuando se modela el negocio y se representa a través de los requerimientos. Los casos de uso guían el proceso de desarrollo ya que los modelos que se obtienen como resultado de los diferentes flujos de trabajo, representan la realización de estos.

Centrado en la arquitectura. La arquitectura en un sistema software se describe mediante diferentes vistas del sistema en construcción. Este concepto incluye los aspectos estáticos y dinámicos más significativos del sistema y se refleja en los casos de uso, pues cada producto tiene tanto una función como una forma, ninguna es suficiente por sí sola.

Iterativo e incremental. Resulta práctico dividir el trabajo en partes más pequeñas o mini proyectos, los cuales no son más que iteraciones que resultan en un incremento. Una iteración es una secuencia de actividades con un plan establecido y criterios de evaluación, cuyo resultado es una versión del software(14).

1.4.2 Lenguajes de modelado UML

Las aplicaciones deben ser estructuradas de una manera que permita la escalabilidad, la seguridad y su ejecución sólida bajo determinadas condiciones, y su arquitectura se debe definir con suficiente claridad para que los programadores puedan encontrar y corregir un error que aparece mucho después de que los autores originales se han trasladado a otros proyectos. El Lenguaje Unificado de Modelado (UML) del inglés (Unified Model Language) es un lenguaje de modelado visual que se usa para

especificar, visualizar, construir y documentar artefactos de un sistema de software, incluida su estructura y diseño, de manera que cumpla con todos los requisitos. (se puede usar UML para el modelado de negocios y modelos de otros sistemas no-software también.) UML está compuesto por diversos elementos gráficos que se combinan para conformar diagramas y cuentan con reglas para combinar tales elementos. Mediante UML es posible establecer la serie de requerimientos y estructuras necesarias para plasmar un sistema de software previo al proceso intensivo de escribir código(15).

1.4.3 Herramienta CASE

Las aplicaciones informáticas que facilitan el trabajo dentro del ciclo de desarrollo del software son conocidas como herramientas CASE (del inglés: Computer Aided Sofware Engineering) Ingeniería de Software Asistida por Computadora y se emplean para aumentar la productividad del desarrollo del software disminuyendo los tiempos de construcción y el costo de los mismos. Como herramienta CASE se escogió Visual Paradigm. Es una herramienta multiplataforma que utiliza UML como lenguaje de modelado. Es muy fácil de usar y presenta un ambiente gráfico agradable para el usuario. Soporta la última notación UML: 2.1, ingeniería inversa, generación de código, generador de informes, editor de figuras, exportación de diagramas a partir de selecciones parciales o completas en JPG, integración con MS Visio, integración IDE con Eclipse, NetBeans y otros. Entre sus nuevas características se incluyen el modelado colaborativo con CVS y Subversión e interoperabilidad con modelos UML2 a través de XML. Incorpora el soporte para trabajo en equipo, lo que permite que varios desarrolladores trabajen a la vez en el mismo diagrama y vean en tiempo real los cambios hechos por sus compañeros (16).

1.4.4 Lenguajes de programación y entornos de desarrollo propuestos

Dentro de las características que debe cumplir el software a elaborar es que este sea multiplataforma y además utilice para su elaboración herramientas gratuitas por lo cual hemos seleccionado dentro de estas las mejores alternativas en la actualidad.

Lenguajes de Programación

Python

Python está bajo una licencia de código abierto que permite su libre uso y distribución, incluso con fines comerciales. Es un lenguaje dinámico de programación muy potente que se utiliza en una amplia variedad de aplicaciones. Python es multiplataforma está disponible para los principales sistemas operativos: Windows, Linux / Unix, OS / 2, Mac, Amiga, entre otros. Es a menudo comparado con Tcl, Perl, Ruby, Scheme o Java. Algunas de sus características distintivas son (17):

- Muy claro, la sintaxis legible
- Introspección
- Orientado a objetos
- Completa modularidad, el apoyo a los paquetes jerárquico
- Manipulación de errores
- Datos dinámicos de alto nivel

- Amplias bibliotecas estándar y módulos de terceros para prácticamente todas las tareas
- Extensiones y módulos fácilmente escrito en c, c + + (o java para jython, o. Net para ironpython)
- Integrable dentro de las aplicaciones como una interfaz de scripting.

Lenguaje C++

La persona a la que se le acredita la creación del C++ es Bjarne Stroustrup. El desarrollo el C++ para tratar de programar simulaciones de eventos sobre las que había estado trabajando años atrás. Stroustrup se dio cuenta que los lenguajes de programación normales, que no eran OOP, no podían manejar las tareas de simulación de los eventos del mundo real tan bien como un lenguaje orientado a objetos.

Stroustrup trabajó con los laboratorios Bell de la AT&T para desarrollar y mejorar el C++ durante varios años. El American Nacional Standards Institute, grupo que norma la mayoría de los lenguajes de computación, ha estandarizado el C++, aunque también el estándar de la AT&T es considerado como el que se debe emular.

A través de los años, AT&T ha aprobado muchas características del C++ que no son OOP pero que hacen al C++ mejor que el C. Tantas mejoras han sido añadidas al C++ que el comité ANSI ha tomado varios elementos del C++ para el lenguaje C. Por ejemplo, los prototipos de función no se iniciaron en el C, aunque han sido parte del lenguaje C desde hace varios años. Los prototipos de función se iniciaron el C++, y, debido a que contribuyen a hacer mejores programas, ANSI incluyó la característica en el lenguaje C (18).

EL **C++** es un lenguaje imperativo orientado a objetos derivado del **C**. En realidad un súper conjunto de **C**, que nació para añadirle cualidades y características de las que carecía. El resultado es que como su ancestro, sigue muy ligado al hardware subyacente, manteniendo una considerable potencia para programación a bajo nivel, pero se la han añadido elementos que le permiten también un estilo de programación con alto nivel de abstracción.

El **C++** no es un lenguaje orientado a objetos puro (en el sentido en que puede serlo Java por ejemplo), además no nació como un ejercicio académico de diseño. Se trata simplemente del sucesor de un lenguaje de programación hecho por programadores (de alto nivel) para programadores, lo que se traduce en un diseño al que se le han ido añadiendo todos los elementos que la práctica aconsejaba como necesarios, con independencia de su belleza o purismo conceptual. Estos condicionantes tienen su cara y su cruz; en ocasiones son motivo de ciertos "reproches" por parte de sus detractores, en otras, estas características son precisamente una cualidad. De hecho, en el diseño de la **Librería Estándar C++** se ha usado ampliamente esta dualidad (ser mezcla de un lenguaje tradicional con elementos de POO), lo que ha permitido un modelo muy avanzado de programación extraordinariamente flexible la **programación genérica**.

En el sistema se escogió C++ debido a ser el lenguaje orientado a objetos más cercano al hardware y además debido a una serie de frameworks que se utilizaran que se desarrollan para este lenguaje de programación.

Entornos de Desarrollo Integrados (IDE)

Propuestas para el Lenguaje C++:

QtCreator **IDE**

QtCreator es un entorno de desarrollo integrado (IDE) multi-plataforma adaptado a las necesidades de los desarrolladores de Qt. Dicho IDE dispone entre sus características (19):

- Editor de código para C++y Java Script
- Diseñador de Interfaz de usuario integrado
- Herramientas de gestión del proyecto y su construcción
- Depuradores de código
- Soporte para el control de versiones

Netbeans

NetBeans es un proyecto exitoso de código abierto con una gran base de usuarios, una comunidad en constante crecimiento. Sun Micro Systems fundó el proyecto de código abierto NetBeans en junio 2000. NetBeans IDE es un entorno de desarrollo una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java pero puede servir para cualquier otro lenguaje de programación incluido C++. Existe además un número importante de módulos para extender el NetBeans IDE. NetBeans IDE es un producto libre y gratuito sin restricciones de uso. Algunas de sus características son (20):

- Sintaxis Resaltada
- Completamiento de Código y Análisis de Tecleo
- Soluciones Rápidas (Quick Fixes) y Verificación de Sintaxis
- Refactorización
- Soporte Java Beans
    - Modelos Bean en el Navegador
    - Generador de Propiedades Bean
    - Editor BeanInfo
- Control de versión ClearCase

- Completamiento de Código Javadoc
    - Soporte de etiquetas (tags) estándares: @param, etc.
- Completamiento de Código para parámetros, excepciones y otros.
- Plantillas de código

Propuestas para el Lenguaje Python:

EasyEclipse con plugin PyDev y QtDesigner

EasyEclipse es una distribución del IDE Eclipse cuyo objetivo es ser más ligera y sencilla. Existe una versión que trae integrado el plugin PyDev llamada EasyEclipse for Python que se ha convertido en uno de los IDEs más populares para dicho lenguaje.

PyDev es un plugin para Eclipse que con los casi todos los requisitos (21):

- Resaltado de código
- Sangrado automática
- Completado automático
- Ejecución de programas
- Depurador de soluciones
- Administrador de proyectos

El lenguaje de programación Python no cuenta con un IDE robusto que sea libre y tenga además interfaz de usuario, EasyEclipse es un IDE que cumple con las primeras condiciones pero no permite crear interfaces de usuario por lo tanto para complementar se debe usar QtDesigner para esta tarea.

QtDesigner es una herramienta de la librería Qt para el diseño y la creación de interfaces gráficas de usuario (GUI) usando los componentes de Qt. Permite crear y personalizar tus widgets[4] o controles, y probarlos utilizando diferentes estilos y resoluciones. Los widgets y formularios creados con QtDesigner se integran a la perfección con el código de programación y permite asignar fácilmente el comportamiento de los elementos gráficos. Todas las propiedades establecidas en QtDesigner se pueden cambiar dinámicamente en el código. Además, características como la promoción widget y plugins personalizados le permiten utilizar sus propios componentes con QtDesigner(22).

[4] Representación estandarizada en pantalla de un control que el usuario puede manipular. Ejemplos de widgets son las barras de desplazamiento, los botones y las cajas de texto.

1.4.5 Software Blender como framework y Herramienta CAD embebida:

CAD significa Diseño Asistido por Computadora, acrónimo del inglés (Computer-Aided Design). Existen dos tipos de herramientas CAD: programas de dibujo en dos dimensiones (2D) y modeladores en tres dimensiones (3D). Los modeladores en 3D añaden superficies y sólidos, estos últimos son los de interés para nuestro proyecto para representar los escenarios interiores donde se simulará el fenómeno de la propagación. El entorno del Blender es completamente libre de código abierto y con el plugin BlenderCAD es un CAD liguero y potente. Está programado en C/C++ y puede ser extendido con Python. Es posible a través de plugins añadirle las funcionalidades necesarias para que cumpla con los requisitos de nuestra aplicación, como son permitir la adición de las propiedades físicas de los materiales necesarios para realizar los cálculos y definir el grosor de las superficies añadidas. Este programa ya posee la capacidad de aplicar un material sobre una superficie y de crear un material, pero está orientado a las propiedades de que afectan a las ondas lumínicas que son similares a las ondas de radio. Con el plugin BlenderCAD permite que este diseñe los objetos de manera más precisa como es necesario en nuestro sistema(23).

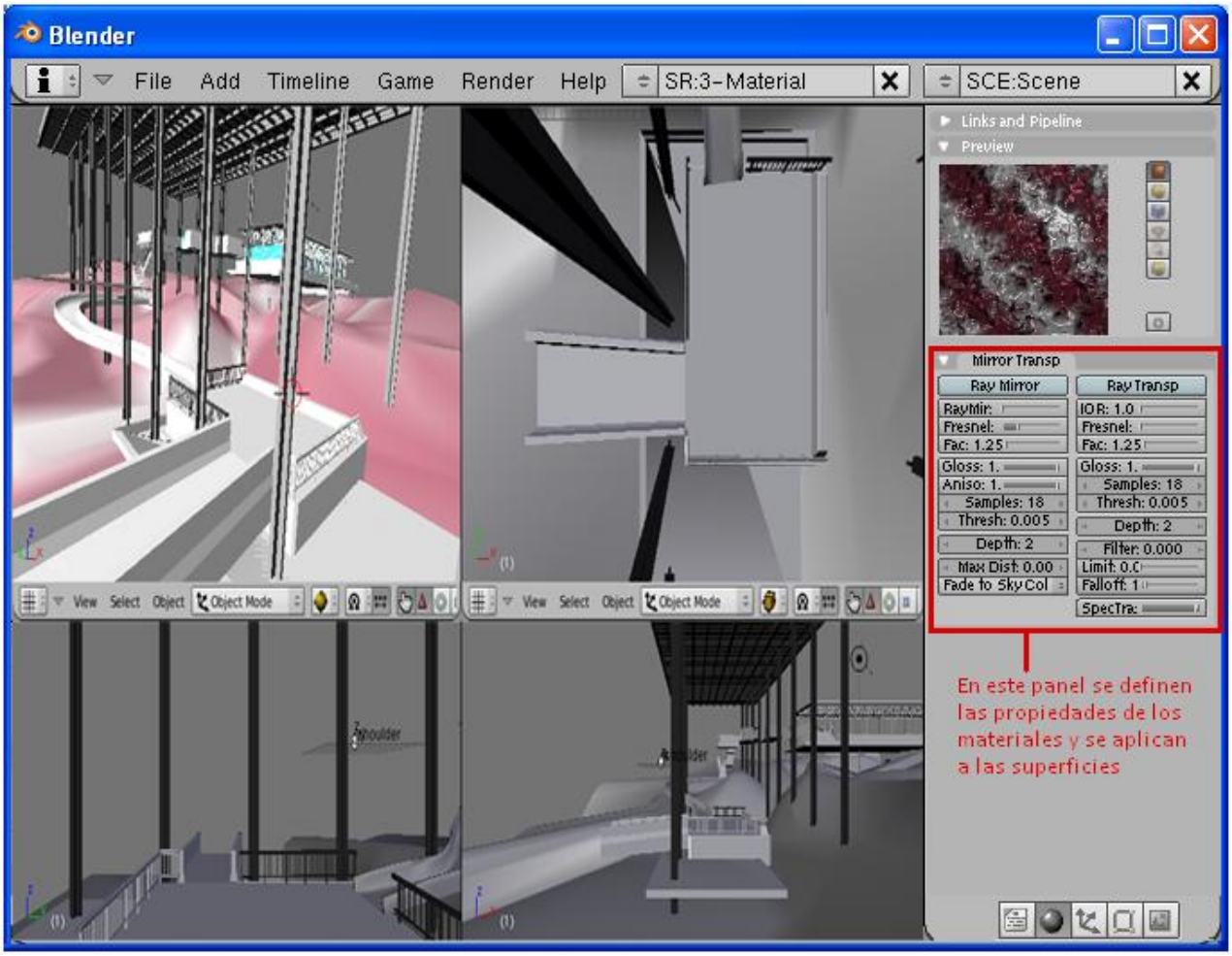


Figura 1.9Interfaz gráfica del programa Blender.

Lista de Herramientas CAD en la actualidad (24):

- Libres:

  - ✓ Sweet Home 3D
  - ✓ Blender con plugin BlenderCAD
  - ✓ FreeCAD
  - ✓ INSTACAD

- Propietarias:

- ✓ Autodesk AutoCAD
- ✓ ArchiCAD
- ✓ 3DVIA Technical Communication Software

Herramientas de Modelado 3D:

- Libres:
  - ✓ Blender
  - ✓ Wings 3d
  - ✓ Google SketchUp
- Propietarias:
  - ✓ Autodesk 3ds Max
  - ✓ Autodesk Maya
- ✓ Lightwave 3D

Blender con plugin BlenderCAD embebido en otro sistema.

El único control del propio sistema operativo en que corra Blender que necesita para ejecutarse es el control ventana ya que él hace uso de la librería OpenGL para mostrar todos los elementos de la GUI que el mismo programa implementa. La librería OpenGL necesita solo un manipulador del componente (el cual está definido en el código del Blender) para representar sus gráficos. Como el código fuente del programa es de libre acceso este programa se puede embeber en el sistema a implementar redirigiendo la GUI de este hacia otro control como pudiera ser un panel figura 1.29 y 1.30.

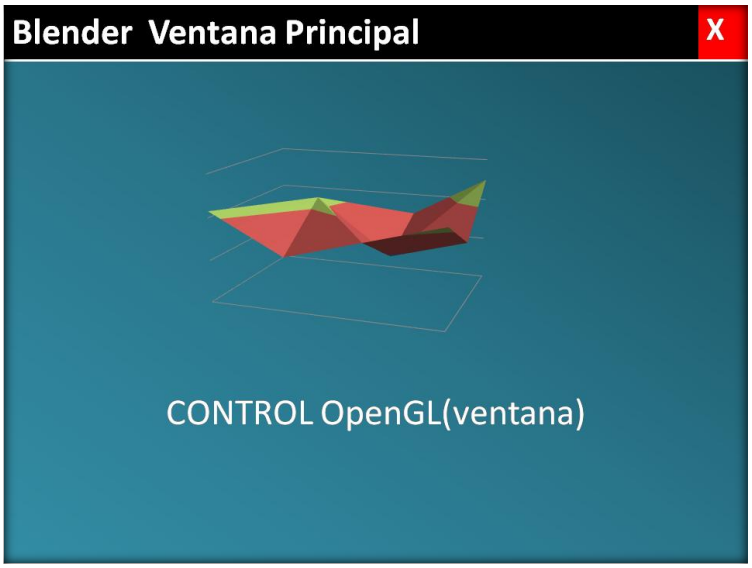


Figura 1.10 Programa Blender Nativo

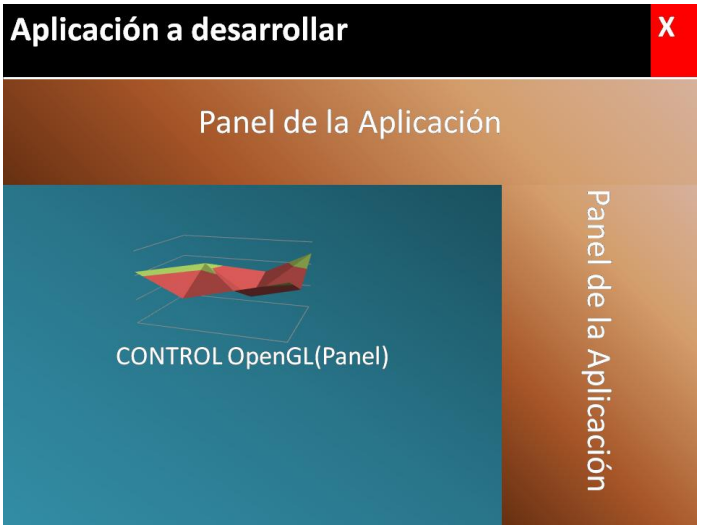


Figura 1.11 Programa Blender Embebido en el sistema a desarrollar

Manejo de los formatos CAD y 3D

Existen diferentes formatos para representar los escenarios tridimensionalmente. Entre los formatos CAD y 3D más populares se encuentran los siguientes:

Formato DXF:

DXF (acrónimo del inglés Drawing Exchange Format) es un formato de archivo informático para dibujos de diseño asistido por computadora, creado fundamentalmente para posibilitar la interoperabilidad entre los archivos .DWG, usados por el programa AutoCAD, y el resto de programas del mercado(25).

Formato OBJ:

OBJ (o. OBJ) es un formato de archivo que almacena geometría tridimensional desarrollado por Wavefront Technologies para su visualizador de animación. El formato de archivo está abierto y ha sido adoptado por otros proveedores de aplicaciones de gráficos 3D. En su mayor parte se trata de un formato universalmente aceptado.

El formato de archivo OBJ es un formato sencillo de datos que representa la geometría 3D, la posición de cada vértice, la posición de UV de cada coordenada de textura de vértices, normales, y las caras que hacen que cada polígono definido como una lista de vértices y vértices de textura. Los vértices se almacenan en un orden a la izquierda de forma predeterminada, por lo que la declaración explícita de las normales innecesarios (26).

Formato 3DS:

3DS es uno de los formatos de archivo utilizados por el software de modelado, animación y render 3D Autodesk 3ds Max 3D. Fue el formato de archivo nativo de la antigua versión Autodesk 3D Studio DOS, que fue muy popular hasta que su sucesor (3D Studio MAX 1.0) lo reemplazó en abril de 1996. Después de surgir en 1990 (cuando la primera versión de 3D Studio DOS se puso en marcha), ha crecido hasta convertirse en un estándar del sector para la transferencia de modelos entre los programas de 3D, o para el almacenamiento de los modelos 3D (27).

Blender como Framework

Reutilizar el código fuente de Blender y embeberlo en el sistema, permite tener de antemano implementado el soporte para una amplia gama de formatos de Escenarios tridimensionales en la figura 1.31 mostramos la lista que el trae por defecto.

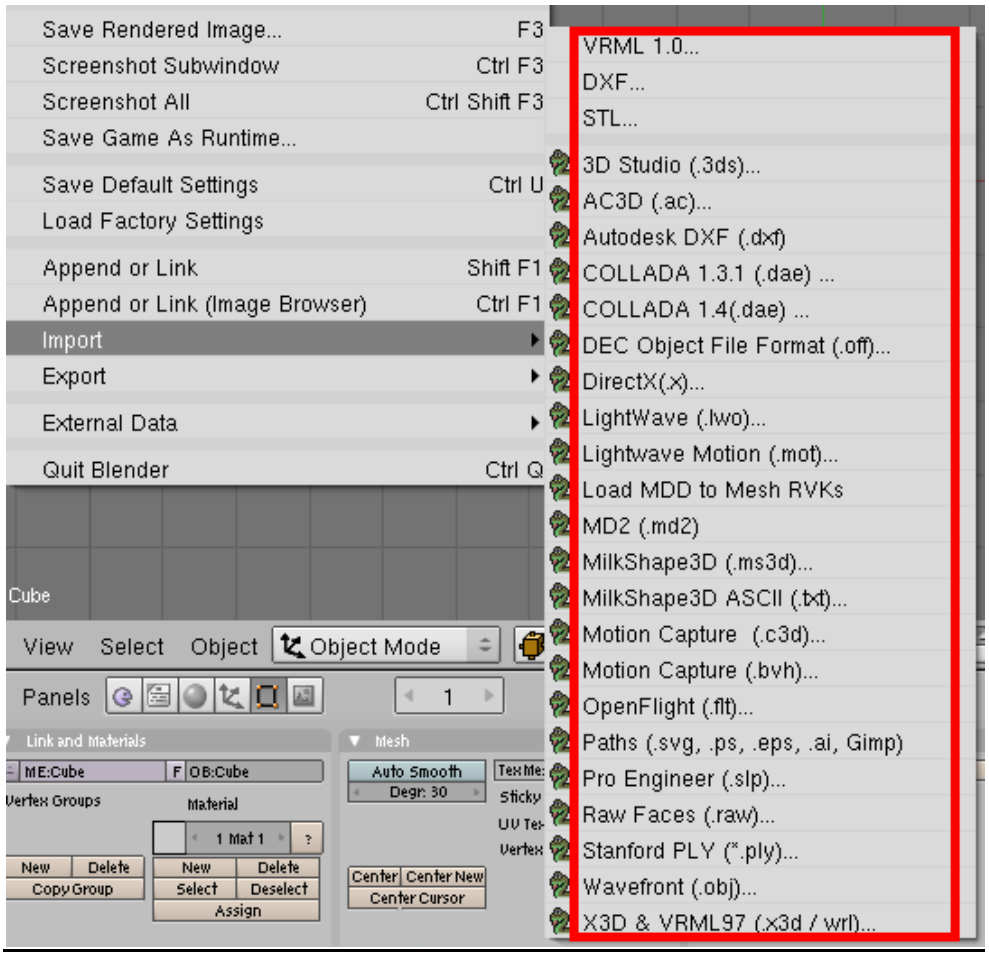


Figura 1.12 Formatos que el Blender es capaz de Importar

Además trae soporte multi-idioma incluido el español. Aplica materiales a los cuerpos, implementa el modelo raytracing ambos con cualidades ópticas y no radioeléctricas problema que es corregible gracias a ser de código abierto.

1.4.6 Framework y librerías propuestas

OpenGL

OpenGL es una librería grafica 3D necesaria para representar los diversos elementos de las antenas y el entorno en donde se simula la propagación. OpenGL fomenta la innovación y el desarrollo de las

aplicaciones mediante la incorporación de un amplio conjunto de prestación, mapeado de texturas, efectos especiales, y otras funciones de visualización de gran alcance.

Cualquier aplicación informática visual que requieren el máximo rendimiento, desde la animación 3D de CAD para la simulación visual de alta calidad pueden explotar las capacidades de alto rendimiento de OpenGL Estas capacidades permiten a los desarrolladores en diversos mercados tales como la radiodifusión, CAD, entretenimiento, imágenes médicas, y la realidad virtual para producir y mostrar gráficos de alta calidad.

Los desarrolladores pueden aprovechar la potencia de OpenGL en todas las plataformas y estaciones de trabajo. No hay otra API gráfica que funcione en una gama tan amplia de plataformas de hardware y entornos de software como OpenGL. No solo se ejecuta en todos los sistemas operativos, incluyendo MacOS, OS/2, UNIX, Windows 95/98, Windows 2000, Windows NT, Linux, OpenStep, y BeOS, sino que también trabaja con todos los sistemas de ventanas populares, incluyendoWin32, MacOS, Presentación Manager, y el sistema X-Windows. OpenGL se puede usar junto con muchos lenguajes de programación incluyendo Ada, C, C++, Fortran, Python, Perl y Java (28).

Framework Qt

Qt es un framework de interfaz de usuario multiplataforma. Utilizando Qt, se puede escribir las aplicaciones una vez e implementar los a través de escritorio, móviles y sistemas operativos integrados sin reescribir el código fuente. Posee una interfaz gráfica de usuario avanzada(Juego completo de controles: botones, cuadros de diálogo, lista jerárquica de árbol, tablas), Auto-escala de las fuentes tipográficas, soporta anti-aliasting y gráficos vectoriales escalables (svg), presenta una personalización completa de la interfaz de usuario con el API de estilo y las hojas de estilo, soporte para hardware de aceleración de gráficos, compatibilidad de gráficos 3d con OpenGL, multithreading, sistema integrado de ventanas, comunicación entre objetos, gráficos 2d, framework de multimedia entre otros (29).

.

Capítulo 2: Descripción y Características del Sistema

Introducción

En el presente capítulo se realiza una descripción del sistema a desarrollar, se ofrecen las características del mismo así como el modelo de dominio correspondiente. Se explican las causas por las cuales se decide realizar Modelo de Dominio en lugar de un modelo de Negocio.

Se enmarca el contexto del sistema, donde se hizo necesario definir los conceptos y cualidades del Modelo de Dominio. Además se especifican los requerimientos que debe tener el sistema, se realizan los diagramas correspondientes para el entendimiento de las funcionalidades del mismo. Se determinan también las propiedades o cualidades que deberá tener para darle cumplimiento a dichas funcionalidades, por lo que se identifican los requerimientos no funcionales.

2.1 Modelo de Dominio

Desde el punto de vista del negocio el problema tiene un bajo nivel de estructuración por lo que se decide crear un Modelo de Dominio en lugar de un Modelo de Negocio. El Modelo de Dominio comprende los conceptos que utilizan los usuarios y con los que deberá trabajar la aplicación. En el este modelo se capturan los objetos más importantes y las relaciones entre estos dentro del entorno donde se desarrollará el sistema. Permite comprensión mutua entre desarrolladores y usuarios finales que ayude en el desarrollo del proyecto a entender el contexto del mismo, para lograr así una captura correcta de requisitos.

Un actor del negocio es cualquier individuo, grupo, entidad, organización, máquina o sistema de información externos; con los que el negocio interactúa. Lo que se modela como actor es el rol que se juega cuando se interactúa con el negocio para beneficiarse de sus resultados.
Un trabajador del negocio representa un rol que juega una persona (o grupo de personas), una máquina o un sistema automatizado; actuando en el negocio. Son los que realizan las actividades, interactuando con otros trabajadores del negocio y manipulando entidades.

Se plantea que el negocio tiene baja estructuración por contar solo con un trabajador del negocio en este caso el ingeniero en telecomunicaciones (el cual será usuario del sistema una vez implementado) y no contar con ningún actor del negocio. El modelo de dominio se muestra a continuación Figura 2.1a y b.

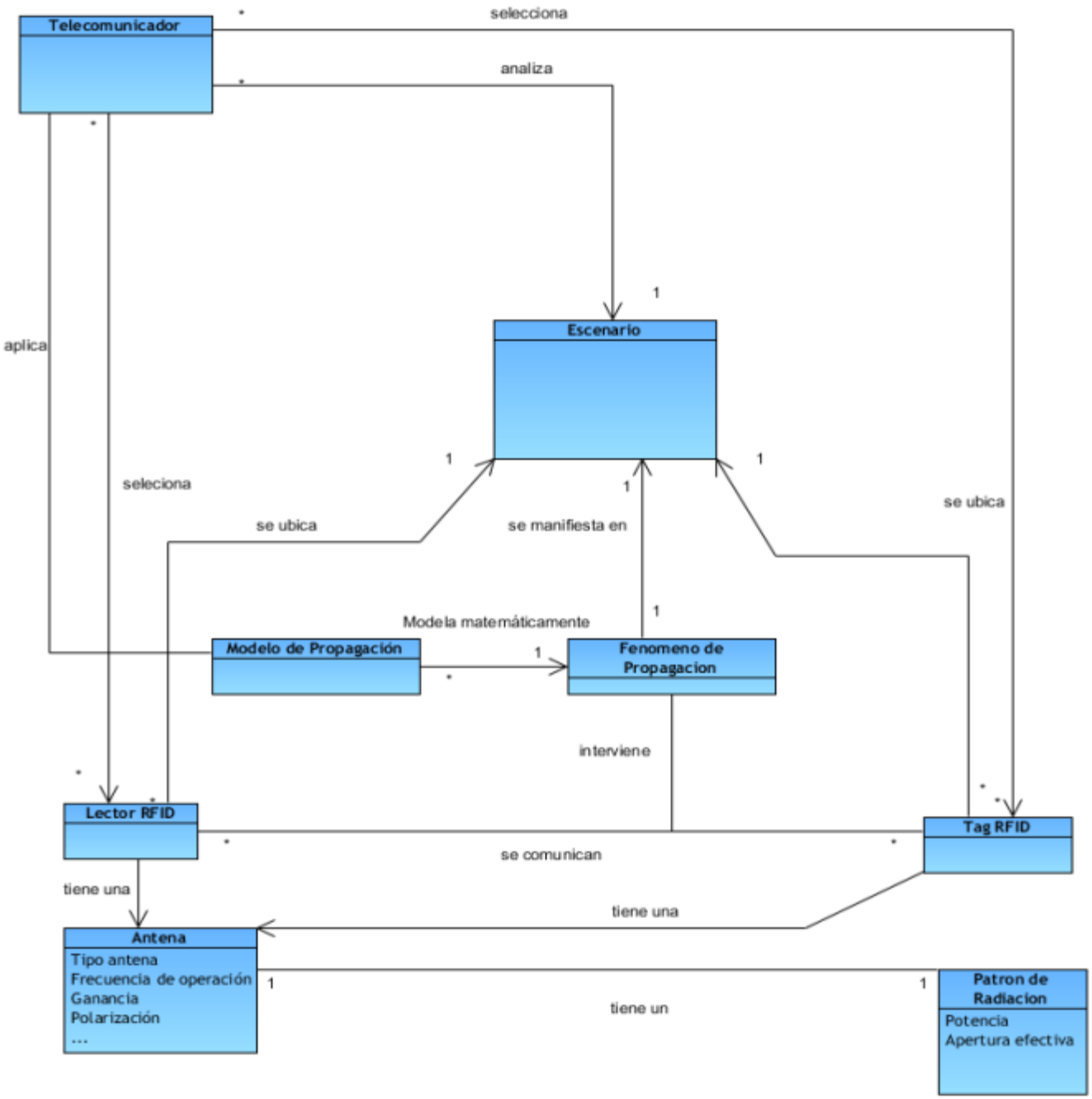


Figura 2.1 a) Modelo del Dominio. Parte 1

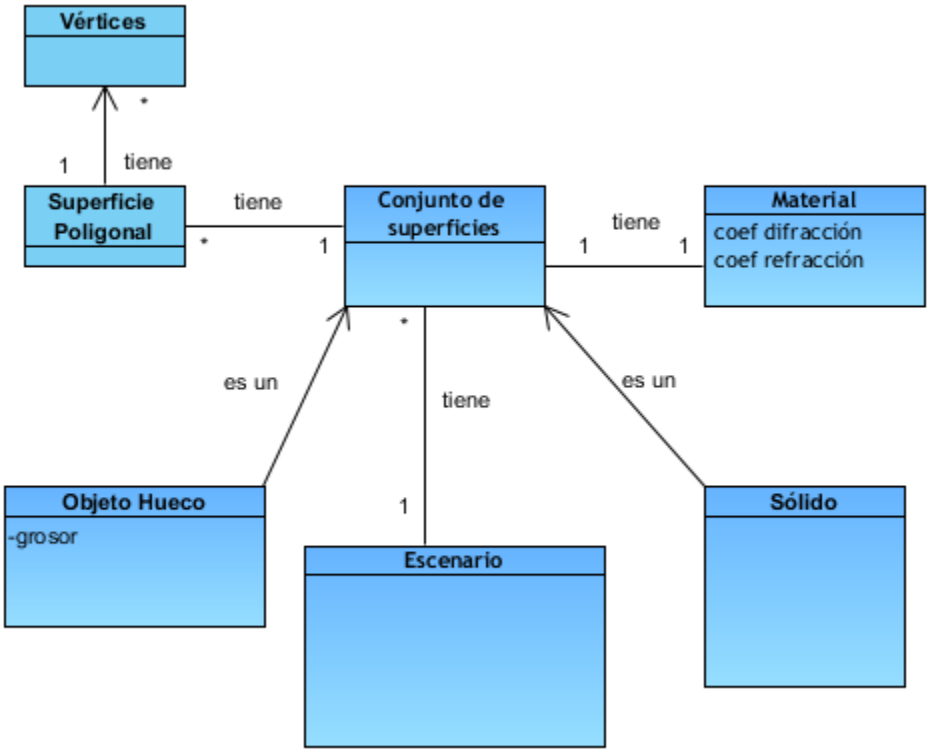


Figura2.1b) Modelo del Dominio. Parte 2

2.1.1 Descripción de los conceptos del Dominio

Para entender mejor el modelo de dominio se explicaran los conceptos principales.

Telecomunicador: Especialista en telecomunicaciones que analiza y estudia el **Escenario**, aplicando modelos de radio propagación, para la selección correcta y ubicación de los componentes necesarios del sistema RFID.

Escenario: Terreno en el cual se despliega el sistema RFID, en este caso el estudio se concibe sólo para ambientes en interiores.

Superficie Poligonal: Define un área en un plano tridimensional delimitada por **Vértices**.

Vértices: Coordenadas (X, Y, Z) en un espacio tridimensional.

Conjunto de Superficies: Agrupación de superficies que definen un objeto en el escenario

Sólido: Cuerpo sólido delimitado por un **Conjunto de Superficies** ejemplo una columna sólida.

Objeto Hueco: Objeto conformado por las superficies y no por lo que ellas encierran en su interior, poseen grosor ejemplo falso techos.

Lector RFID: Dispositivo que localiza y analiza los datos de las **Etiquetas RFID** (Véase en epígrafe 1.4).

Etiqueta RFID: Dispositivo que se inserta en el objeto que se desea tener localizado, para conocer sus parámetros (Véase epígrafe 1.4).

Antena: Componente activo, diseñado para transmitir o recibir ondas electromagnéticas (Véase en epígrafe 1.6).

Patrón de radiación: Zona tridimensional irradiada alrededor de la antena (Véase en epígrafe 1.6.3).

Fenómeno de Radio Propagación: Anomalías propias del canal de radio que afectan la comunicación entre los dispositivos transmisores y receptores, en este caso entre el **Lector RFID** y las **Etiquetas RFID**.

Modelo de Radio Propagación: Modelo matemático que estudia el **Fenómeno de Radio Propagación**, con el cual se obtienen resultados aproximados de estas anomalías para su posterior análisis.

2.2 Levantamiento de Requisitos

El propósito principal del flujo de trabajo Levantamiento de Requisitos es guiar el desarrollo hacia un sistema correcto. Lograr un entendimiento común entre los usuarios y el equipo de proyecto sobre lo que hay que hacer es la clave del éxito en la producción de un software.

Un requisito es una condición o capacidad que tiene que ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar, u otro documento impuesto formalmente. Todas las ideas que los clientes, usuarios y miembros del equipo de proyecto tengan acerca de lo que debe hacer el sistema, deben ser analizadas como candidatas a requisitos. Los requisitos se pueden clasificar en: funcionales y no funcionales.

2.2.1 Requisitos Funcionales

Los requerimientos funcionales son condiciones o capacidades que el sistema debe cumplir. Garantiza darles respuesta de forma satisfactoria a los usuarios finales del mismo, en fin definen lo que el sistema debería hacer. Son declaraciones de los servicios que debe proporcionar el sistema. Especifican la manera en que éste debe reaccionar a determinadas entradas. Especifican cómo debe comportarse el sistema en situaciones particulares.

Los requerimientos funcionales son acciones fundamentales que deben tener lugar en el software al recibir información, procesarla y producir resultados. Su definición debe ser clara y libre de ambigüedades. Estos describen lo que el sistema debe hacer.

RF 1: Gestionar el escenario virtual: Hace referencia a crear y modificar, eliminar y guardar en disco.

RF 2: Gestionar Conjunto de Superficies: Hace referencia a la acción de crear, modificar y eliminar los cuerpos sólidos del escenario.

RF 2.1: Asignar los materiales de un catálogo de materiales a los diferentes objetos del escenario: Hace referencia a que para asignar un material a un cuerpo sólido, se brinda la posibilidad de seleccionar de un catálogo que contenga los materiales creados y los principales materiales encontrados en los escenarios reales.

RF 2.2: Importar objetos al escenario virtual: Hace referencia a la acción de importar al programa un archivo de formatos CAD o 3D externo diseñado en otro programa CAD o 3d.

RF 3: Conversión entre objetos huecos, cuerpo sólido y conjunto de superficies: Hace referencia a que se pueda convertir una de las de las tres entidades nombradas a otra siempre que fuese valido.

RF 4: Gestionar superficies: Hace referencia a la acción de crear, modificar y eliminar las superficies del escenario.

RF 5: Gestionar los materiales virtuales: Hace referencia a crear, modificar y eliminar los materiales con sus propiedades físicas.

RF 6: Gestionarlos patrones de radiación: Hace referencia a crear, modificar y eliminar patrones, usando en su creación diferentes técnicas.

RF 6.1: Generar patrones mediante funciones matemáticas: Hace referencia al uso de funciones matemáticas que modelan el patrón como representación del patrón tridimensional.

RF 6.2: Generar patrones mediante interpolación lineal: Hace referencia al uso de las imágenes bidimensionales vertical y horizontal del patrón (brindadas por muchos proveedores de antenas) para generar un patrón tridimensional.

RF 6.3: Importar modelo tridimensional patrones en los formatos 3D soportados: Hace referencia a importar el patrón desde un formato de archivo 3D soportado.

RF 7: Visualizar Patrones de radiación: Hace referencia a mostrar el patrón y sus características desde diferentes puntos de vista.

RF 7.1: Representar la gráfica 3D del patrón de radiación de la antena: Hace referencia a mostrar el grafico tridimensional del patrón permitiendo cambiar la posición de observación.

RF 7.2: Mostrar los patrones de antena vertical, horizontal y frontal: Hace referencia a mostrar las secciones bidimensionales del patrón (vertical y horizontal).

RF 8: Gestionarlas antenas virtuales con sus diferentes parámetros: Hace referencia a crear las antenas virtuales definiendo los parámetros que permiten observarla como una caja negra asignarle un patrón de los patrones de radiación creados, a modificar las antenas y eliminarlas.

RF 9: Orientar las antenas de Lectores y Etiquetas RFID: Hace referencia a establecer la dirección hacia donde están orientadas las antenas de los lectores y etiquetas RFID.

RF 9.1: Insertar Lectores y Etiquetas RFID desde de un catálogo: Hace referencia a que para insertar tanto un lector como una etiqueta RFID se bride la posibilidad de seleccionar de un catálogo que contenga de ellos los principales que se hallan en el mercado según su aplicación y según su proveedor.

RF 10: Gestionar las Etiquetas RFID: Hace referencia a la acción de crear las Etiquetas RFID, posicionarlas, asignarle una antena, definir si son activas o pasivas, modificarlas una vez creadas y eliminarlas.

RF 11: Gestionar los Lectores RFID: Hace referencia a la acción de crear los Lectores RFID, posicionarlos, asignarle una antena, modificarlas una vez creadas y eliminarlas.

RF 12: Realizar la simulación: Una vez desplegados todos los elementos del escenario, hace referencia a la representación aproximada del comportamiento de la radio propagación muy cercana a lo que ocurre en la realidad.

RF 12.1: Seleccionar y aplicar un modelo de radio propagación: Hace referencia a dar la posibilidad de seleccionar el modelo de radio propagación y realizar los cálculos de simulación.

RF 12.2: Mostrar los resultados de la simulación: Hace referencia a la revelación de los resultados en diferentes vistas para apoyar el análisis del especialista en telecomunicaciones.

2.2.2 Requisitos no Funcionales

Los requerimientos no funcionales son propiedades o cualidades que debe tener el sistema. Establecen los servicios que se espera que el sistema proporcione y las restricciones bajo las que funcionará. Son aquellas características que permiten que el producto final sea más o menos usable, rápido, o qué software o hardware es necesario para que funcione. Pueden agruparse de acuerdo a diversas clases como rendimiento, usabilidad, seguridad, software o hardware.

Para satisfacer al cliente y lograr una buena calidad en el sistema se listaron las siguientes propiedades y cualidades:

Portabilidad:

Necesidad de que el sistema sea multiplataforma.

Usabilidad:

Los especialistas requerirán preparación mínima para interactuar con el módulo. Necesita ser sencillo de usar y presentar una interfaz intuitiva.

Soporte:

Se debe entregar junto a la aplicación un manual de usuario donde se especifiquen como trabajar con el software,

Rendimiento:

El sistema realizara cálculos complejos, por lo que se hace necesario que la velocidad de procesamiento de la información sea rápida.

Restricciones de diseño:

Se debe usar C++ y Python como lenguaje de programación. Como entorno integrado de desarrollo NetBean y QTCreator; UML como lenguaje de modelado y como herramienta CASE para el modelado de los artefactos Visual Paradigm.

Requisitos de Licencia:

Todas las licencias son GPL (Licencia General Publicas), variantes o licencias compatibles con esta.

Software:

El software debe ser multiplataforma, distribuciones de Linux y Microsoft Windows XP o superior.

Hardware:

- Periféricos: Mouse y Teclado.
- Procesador Pentium III o superior a 1 GHZ
- 512 MB de RAM.
- Resolución de pantalla 1024 x 768 pixeles
- Profundidad de color de 16 bit
- Mouse de 3 Botones
- Open GL Graphics Card with 64 MB RAM
- 100Mb de espacio en disco duro.

2.3 Modelo de Casos de Uso del Sistema.

El modelo de casos de uso permite que los desarrolladores del software y los clientes logren un acuerdo sobre lo que el sistema debe hacer, es decir, sobre las condiciones y posibilidades que debe cumplir el sistema. El modelo de casos de uso sirve como convenio entre clientes y desarrolladores.

En el presente epígrafe se identifican los actores así como los casos de uso del sistema, y se determina el “Diagrama de Casos de Uso del Sistema”, que no es más que la representación gráfica a los procesos y su interacción con los actores, se describen además los Casos de Uso del Sistema.

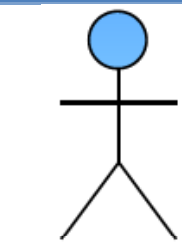
2.3.1 Actores del Sistema

Un actor no es parte del sistema, es un rol de un usuario que puede intercambiar información o puede ser un recipiente pasivo de información y representa a un ser humano, a un software o a una máquina que interactúa con el sistema.

Cada trabajador del negocio, que tiene actividades a automatizar es un candidato a actor del sistema. Si algún actor del negocio va a interactuar con el sistema, entonces también será un actor del sistema.

Los actores del sistema no son parte de él pero pueden intercambiar información con él. Pueden ser un recipiente pasivo de información e incluso representar el rol que juega una o varias personas, un equipo o un sistema.

Tabla 2.1: Definición de los Actores del Sistema

Actores del Sistema	
 Telecomunicador	Especialista en telecomunicaciones, modela el Escenario en el software posiciona e incorpora los elementos del sistema RFID en este, modifica diversos parámetros de las Etiquetas y los Lectores RFID y realiza la simulación para lograr un aproximado de lo que sucede en escenarios reales.

2.3.2 Casos de Uso del Sistema.

El caso de uso es una técnica para capturar información de cómo un sistema o negocio trabaja, o de cómo se desea que trabaje. No pertenece estrictamente al enfoque orientado a objeto, es una técnica para captura de requisitos de software. Los casos de uso describen bajo la forma de acciones y reacciones el comportamiento de un sistema desde el punto de vista del usuario. Permiten definir los límites del sistema y las relaciones entre este y el entorno. Son descripciones de la funcionalidad del sistema independientes de la implementación. Los casos de uso describen qué hace el sistema, no cómo lo hace, por lo que en su modelación se hace necesario tener en cuenta la separación de los objetivos entre las vistas externas e internas.

Un caso de uso puede ser definido como una secuencia de acciones, incluyendo variaciones, que el sistema puede ejecutar y que produce un resultado observable de valor para un actor que interactúa con el sistema.

Los casos de uso son fragmentos de funcionalidad aportan un resultado de valor para sus actores, son artefactos narrativos que describen el comportamiento del sistema. Establece un acuerdo entre clientes y desarrolladores.

A partir de las funcionalidades que el sistema debe cumplir y los requisitos funcionales analizados se identificaron los siguientes casos de usos:

Figura2.3: Diagrama de Casos de Uso del Sistema.

Tabla 2.2: Definición de los Casos de Uso del Sistema

CU1	**Gestionar Escenario**
Actor	Telecomunicador
Referencia	RF1
Descripción	Acción de Crear, Modificar y Eliminar un Escenario.
Descripción	Acción de Crear, Modificar y Eliminar las Superficies
CU2	**Gestionar Conjunto de Superficies**
Actor	Telecomunicador
Referencia	RF2, RF2.1, RF2.2
Descripción	Acción de Crear, Modificar y Eliminar las Superficies.
CU2.1	**Aplicar Material**
Actor	Telecomunicador
Referencia	RF5
Descripción	Aplicar un material a un Conjunto de Superficies y objetos derivados (Sólidos y objetos huecos) del Escenario.
CU3	**Convertir a cuerpo sólido**
Actor	Telecomunicador

Referencia	RF3
Descripción	Convierte un conjunto de superficies o un objeto hueco en cuerpo sólido.
CU4	**Convertir a conjunto de superficies**
Actor	Telecomunicador
Referencia	RF3
Descripción	Convierte un cuerpo sólido u objeto hueco en un conjunto de superficies.
CU5	**Convertir a objeto hueco**
Actor	Telecomunicador
Referencia	RF3
Descripción	Convierte un cuerpo sólido o un conjunto de superficies en un objeto hueco aplicándole un grosor a las superficies.
CU6	**Gestionar Material**
Actor	Telecomunicador
Referencia	RF5
Descripción	Acción de Crear, Modificar y Eliminar los materiales.
CU7	**Gestionar Antena**
Actor	Telecomunicador

Referencia	RF8
Descripción	Acción de Crear, Modificar y Eliminar las antenas.
CU8	**Gestionar Patrón De Radiación**
Actor	Telecomunicador
Referencia	RF6, RF6.1, RF6.2, RF6.3, RF7, RF7.1, RF7.2
Descripción	Acción de Crear, Modificar, Mostrar y Eliminar los Patrones de Radiación.
CU9	**Gestionar Etiqueta RFID**
Actor	Telecomunicador
Referencia	RF10
Descripción	Acción de Crear, Modificar y Eliminar las Etiquetas RFID.
CU10	**Gestionar Lector RFID**
Actor	Telecomunicador
Referencia	RF11
Descripción	Acción de Crear, Modificar y Eliminar los Lectores RFID
CU11	**Seleccionar los modelos de propagación**
Actor	Telecomunicador
Referencia	RF12, RF12.1

Descripción	Selecciona los modelos de propagación a aplicar en la simulación y valida si existen suficientes datos para aplicar estos modelos .
CU12	**Realizar Simulación**
Actor	Telecomunicador
Referencia	RF12,RF 12.2
Descripción	Se efectúa la simulación del fenómeno de radio propagación de sistemas RFID para interiores y se muestran los resultados tangibles de dicho proceso.

Conclusiones

En el presente capítulo se puede observar como a partir de los procesos del dominio se identificaron los requisitos del software a construir, también se puede ver como los caso de uso del sistema se forman agrupando requisitos funcionales, los cuales deben ser verificados y validados para evitar errores. Se describió también la interacción de los actores del sistema con los casos de uso. Se validó correctamente los requisitos con el usuario final mostrándole las funcionalidades del programa a través de la interfaz grafica de usuario.

Capítulo 3: Análisis y Diseño del Sistema.

3.1 Introducción

En este capítulo tiene como objetivo analizar los requisitos funcionales descritos en el capitulo anterior, refinarlos y estructurarlos para obtener una comprensión más certera de los mismos. Se expondrán todos los detalles descriptivos de la solución propuesta. Se obtendrá a través del análisis una visión del sistema propuesto para la realización del simulador para radioenlace RFID.

Permitirá además comprender en profundidad aspectos relacionados con los requisitos no funcionales, lenguajes de programación, componentes reutilizables, tecnologías. Se examinara la arquitectura del sistema, la comunicación y el acoplamiento entre las diferentes capas que lo componen.

3.2 Modelo de Clases del Análisis

El modelo de análisis ofrece un poder expresivo y una formalización para describir los aspectos del sistema, proporciona una estructura centrada en el mantenimiento, específicamente la flexibilidad ante los cambios y la reutilización. El modelo de análisis puede considerarse una primera aproximación al modelo de diseño aunque es un modelo por sí mismo. Es importante también hacer notar que en el modelo de análisis se hacen abstracciones para evitar resolver algunos problemas que es mejor posponer al diseño y la implementación.

Aunque en el modelo del análisis hay un refinamiento de los requisitos, no se tiene en cuenta el lenguaje de programación a usar en la construcción, la plataforma en la que se ejecutará la aplicación, los componentes prefabricados o reusables de otras aplicaciones, entre otras características que afectan al sistema, porque el objetivo del análisis que se realiza es comprender perfectamente los requisitos del software y no precisar cómo se implementará la solución.

3.2.1 Diagrama de Clases de Análisis

Representan un modelo conceptual temprano que describe las características y comportamiento comunes de un conjunto de elementos que existen en el sistema. Se expresa que es conceptual pues pospone todos los elementos de diseño ya que no considera posibles tecnologías a emplear en el desarrollo del software; constituyen un prototipo de las futuras clases que darán vida al mismo. Las clases del Análisis están siempre identificadas con uno de los tres estereotipos existentes, los cuales son:

- ✓ Interfaz: Se encargan de la modelación de toda la interacción que puede existir entre los actores y el sistema; constituyen las fronteras del mismo.
- ✓ Control: Representan la coordinación, secuenciación, transacciones y a veces la lógica del negocio; se emplean a menudo para encapsular el control referido a un CU.
- ✓ Entidad: representa la información de larga duración y a menudo persistente que se maneja en el sistema.

A continuación le mostramos todos los diagramas de clases del análisis por cada Caso de Uso correspondiente a la investigación:

Diagrama de Clases del Análisis del CU Gestionar Escenario

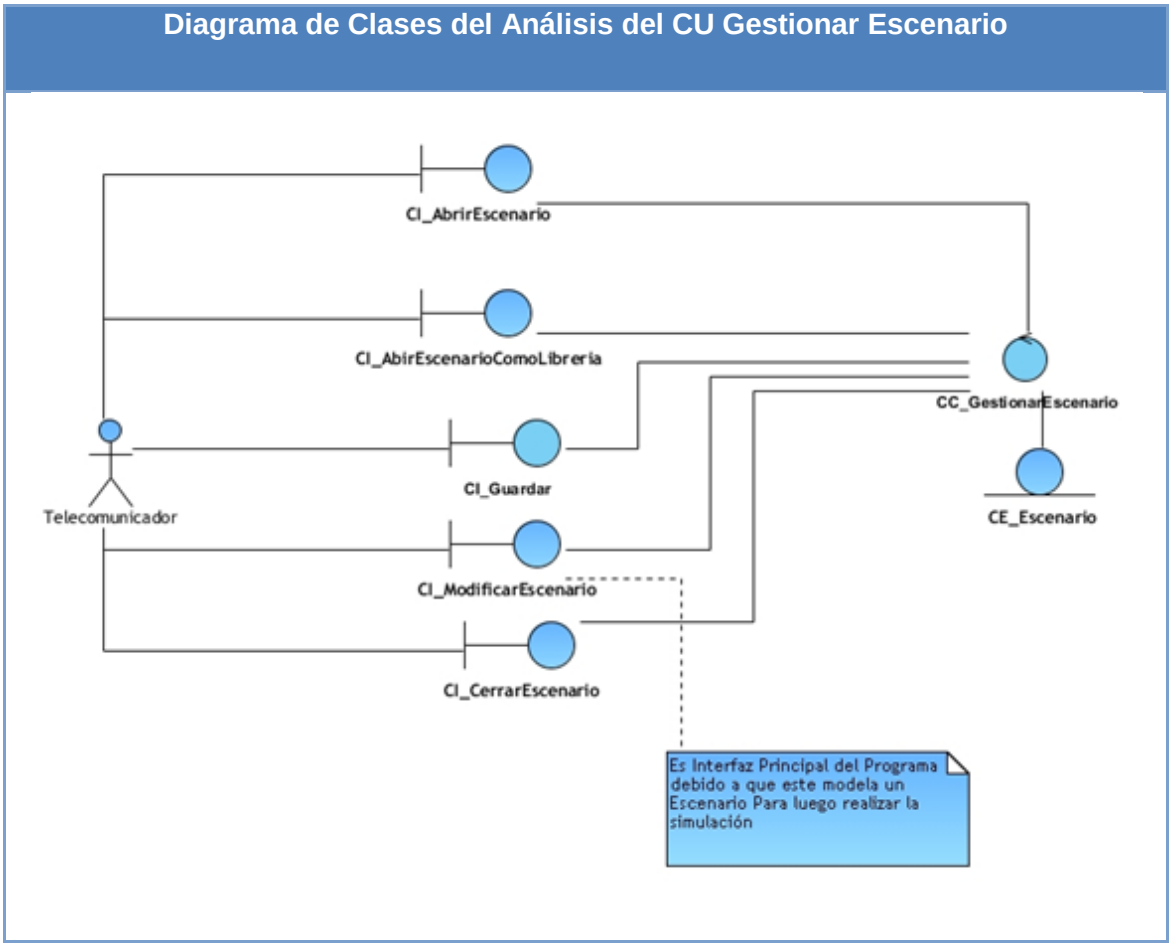


Diagrama de Clases del Análisis del CU Gestionar Conjunto de Superficies y Aplicar Material

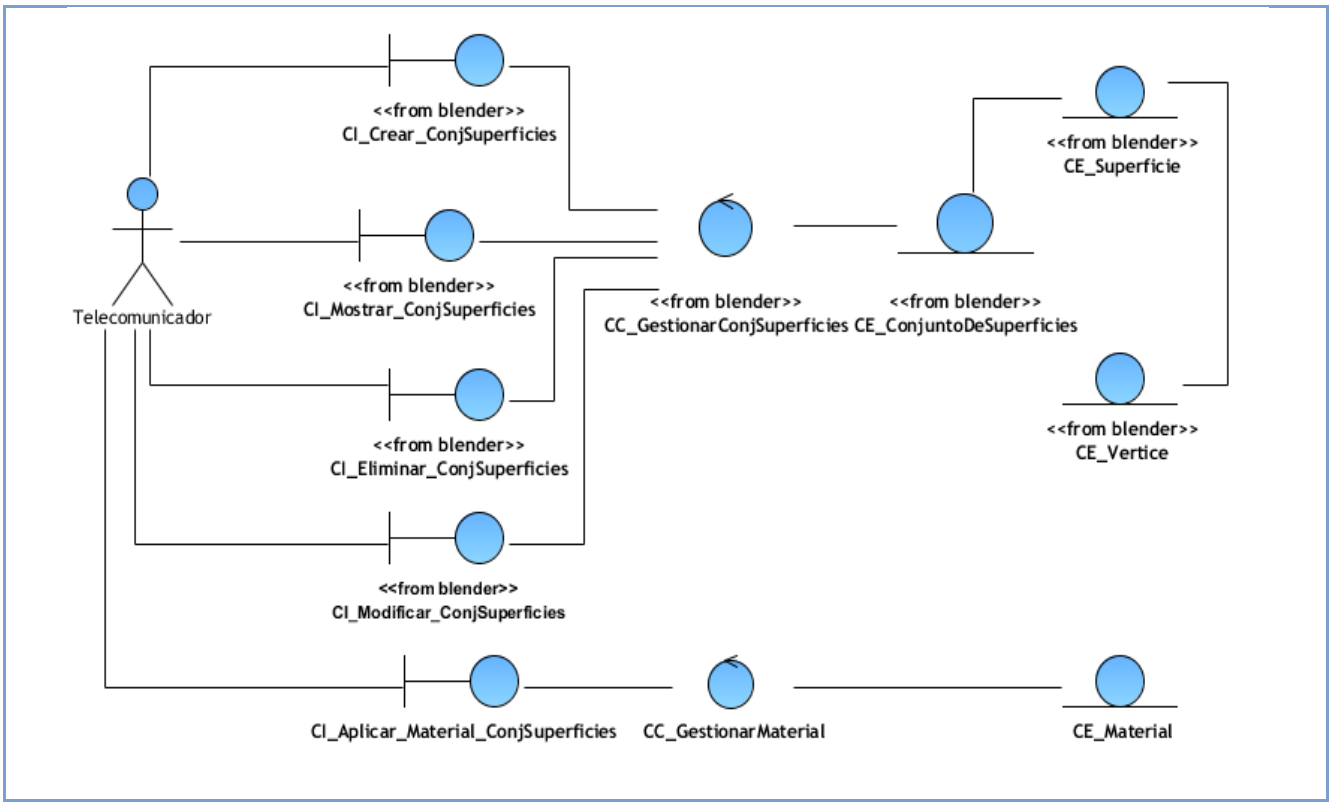


Diagrama de Clases del Análisis del CU Convertir a cuerpo sólido, CU Convertir a conjunto de superficies y CU Convertir a objeto hueco

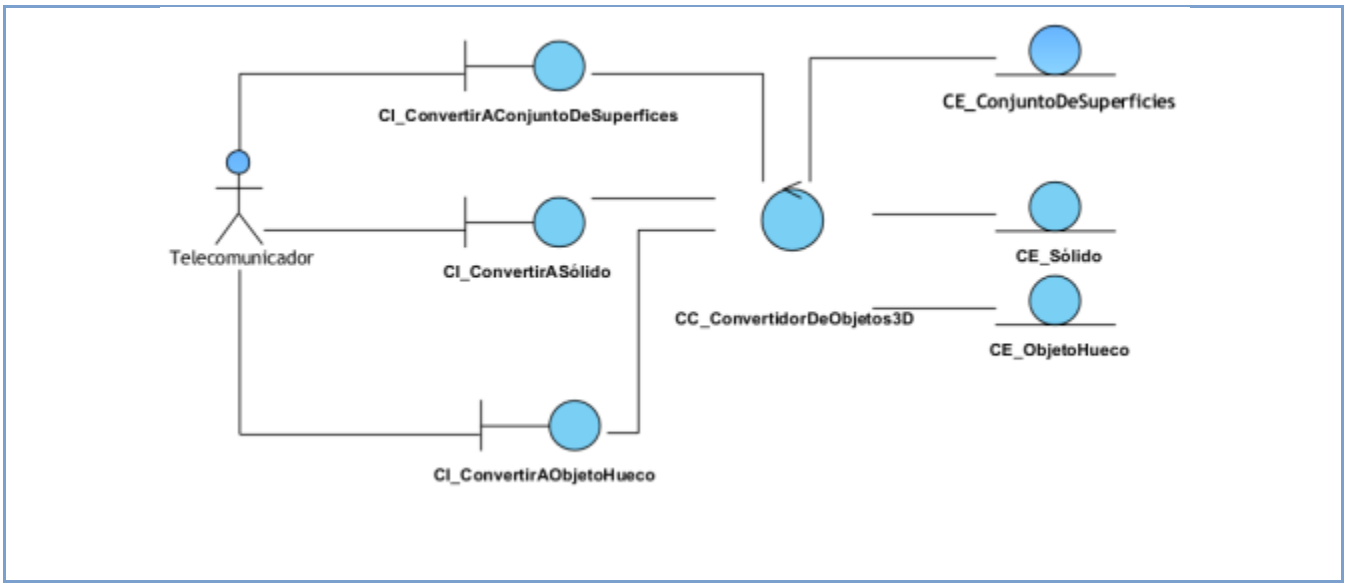


Diagrama de Clases del Análisis del CU Gestionar Material

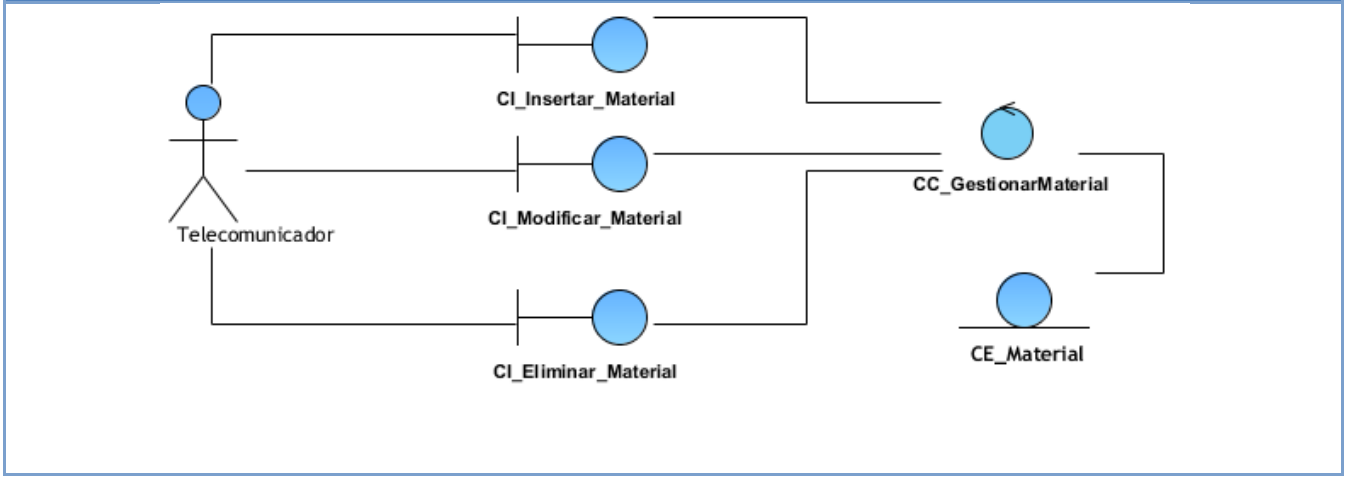


Diagrama de Clases del Análisis del CU Gestionar Antena

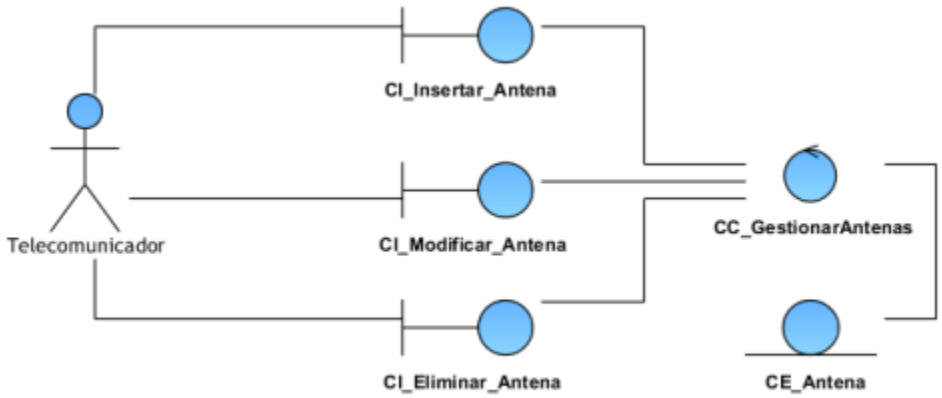


Diagrama de Clases del Análisis del CU Gestionar Patrón De Radiación

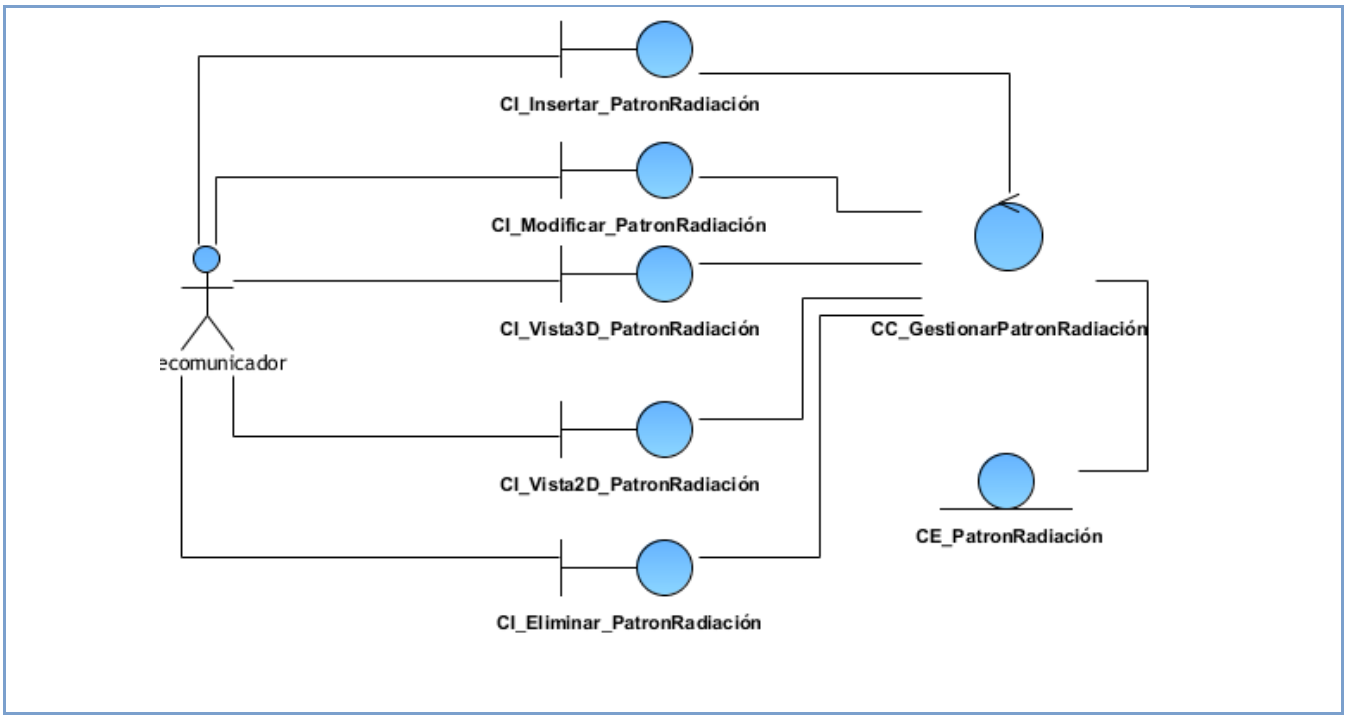


Diagrama de Clases del Análisis del CU Gestionar Etiqueta RFID

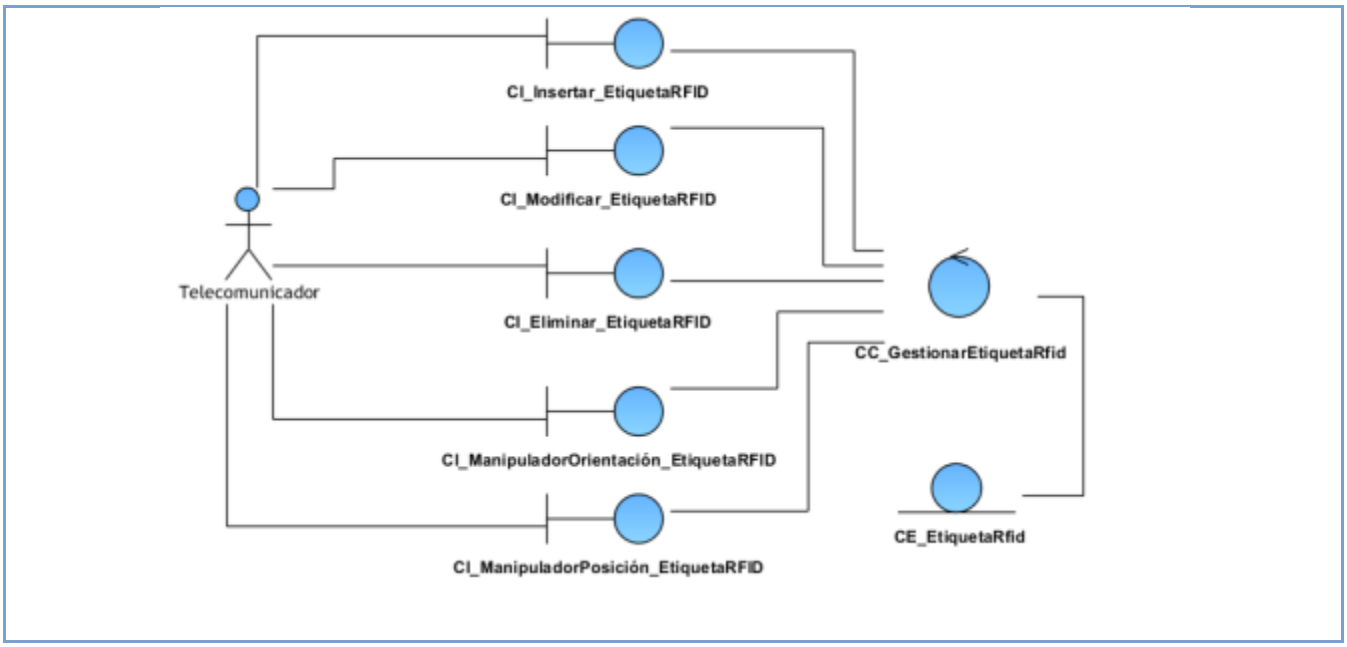


Diagrama de Clases del Análisis del CU Gestionar Lector RFID

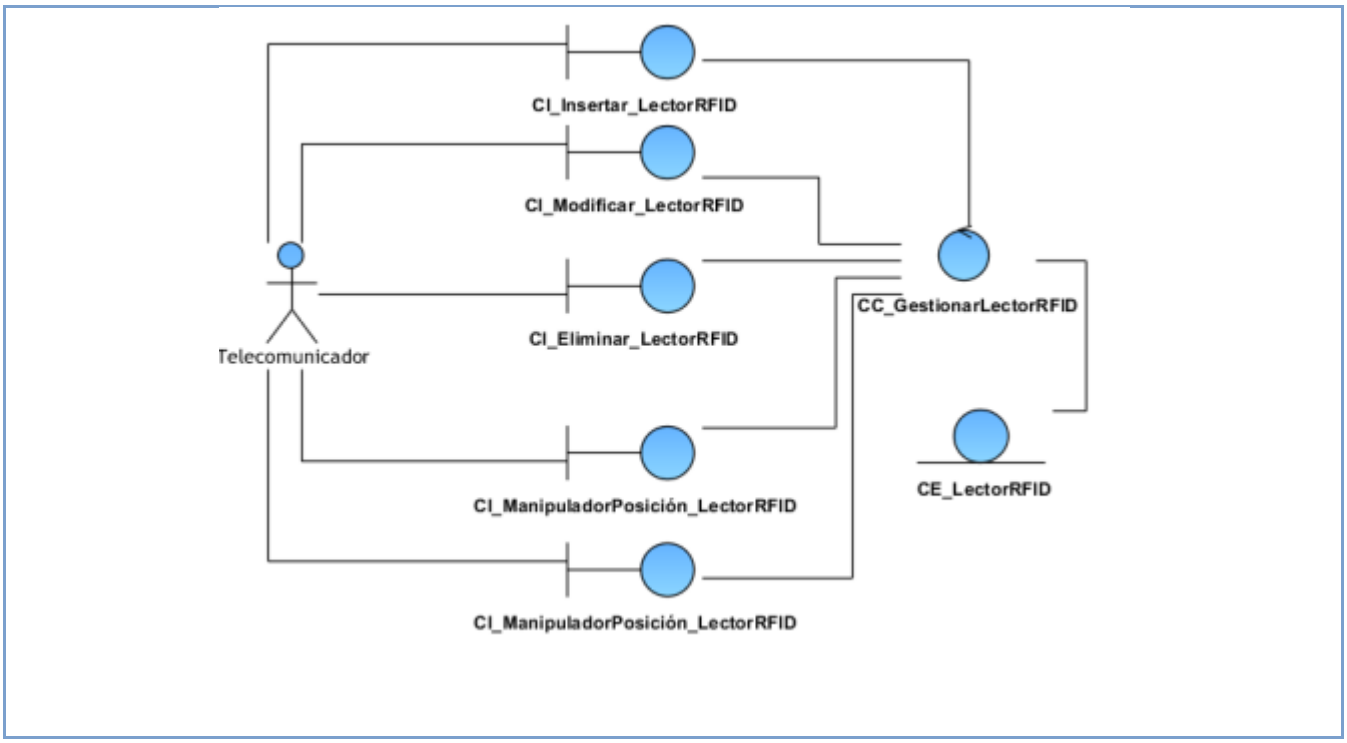


Diagrama de Clases del Análisis del CU Seleccionar los modelos de propagación

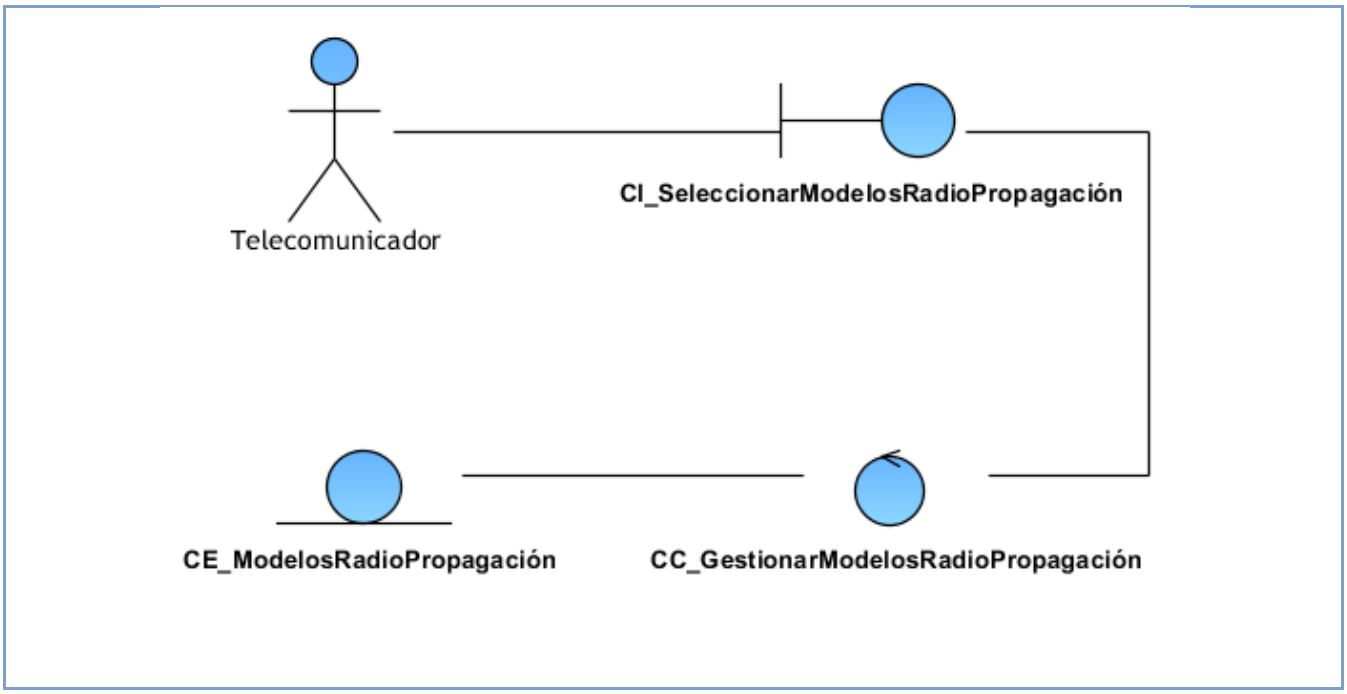
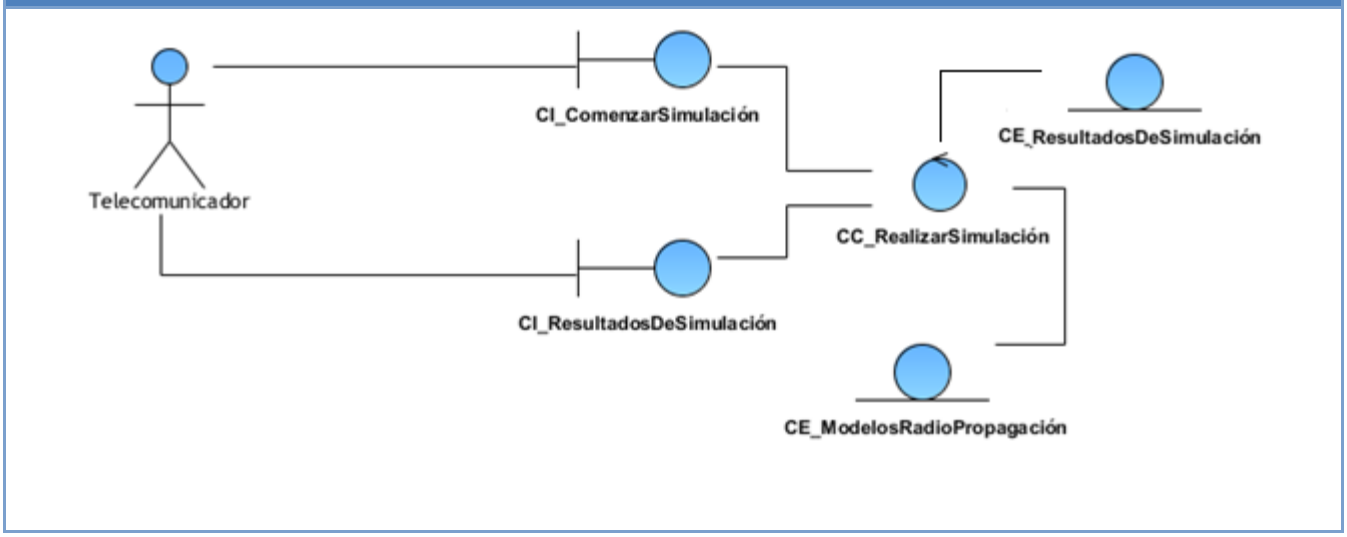


Diagrama de Clases del Análisis del CU Realizar Simulación

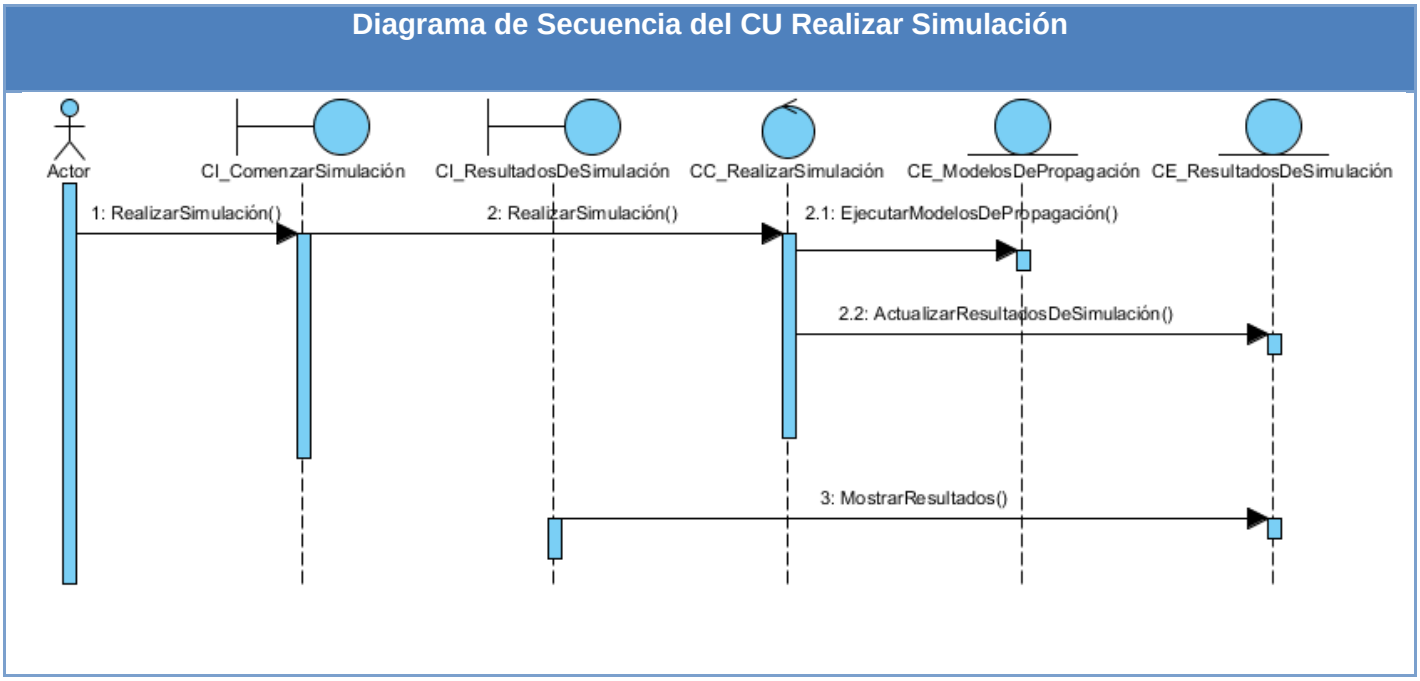


3.2.2 Diagramas de Interacción del Análisis

Los diagramas de interacción se usan para modelar los aspectos dinámicos del sistema, contienen objetos, enlaces y mensajes. Tienen como objetivo describir cómo interactúan las clases con sus respectivos modelos de análisis, mediante diagramas de colaboración o secuencia.

A continuación se muestra el diagrama de secuencia del análisis del caso de uso crítico **Realizar Simulación** correspondiente al sistema:

Diagrama de Secuencia del CU Realizar Simulación



3.3 Modelo de Clases del Diseño

El Modelo de Diseño es un modelo físico ya que es un plano de la implementación, creado principalmente como programación visual. En el modelo de diseño los casos de uso son realizados por las clases del diseño y sus objetos.

En el diseño se modela el sistema y se encuentra su forma, para que soporte todos los requisitos, incluyendo los no funcionales y las restricciones que se le suponen. Una entrada esencial en el diseño es el resultado del análisis, o sea el modelo de análisis mostrado en el epígrafe anterior, que proporciona una comprensión detallada de los requisitos.

El diseño tiene el propósito de formular los modelos que se centran en los requisitos no funcionales y en el dominio de la solución y que prepara para la implementación y prueba del sistema. Pretende crear un plano del modelo de implementación, por lo que el grueso del esfuerzo está en las últimas iteraciones de elaboración y las primeras de construcción.

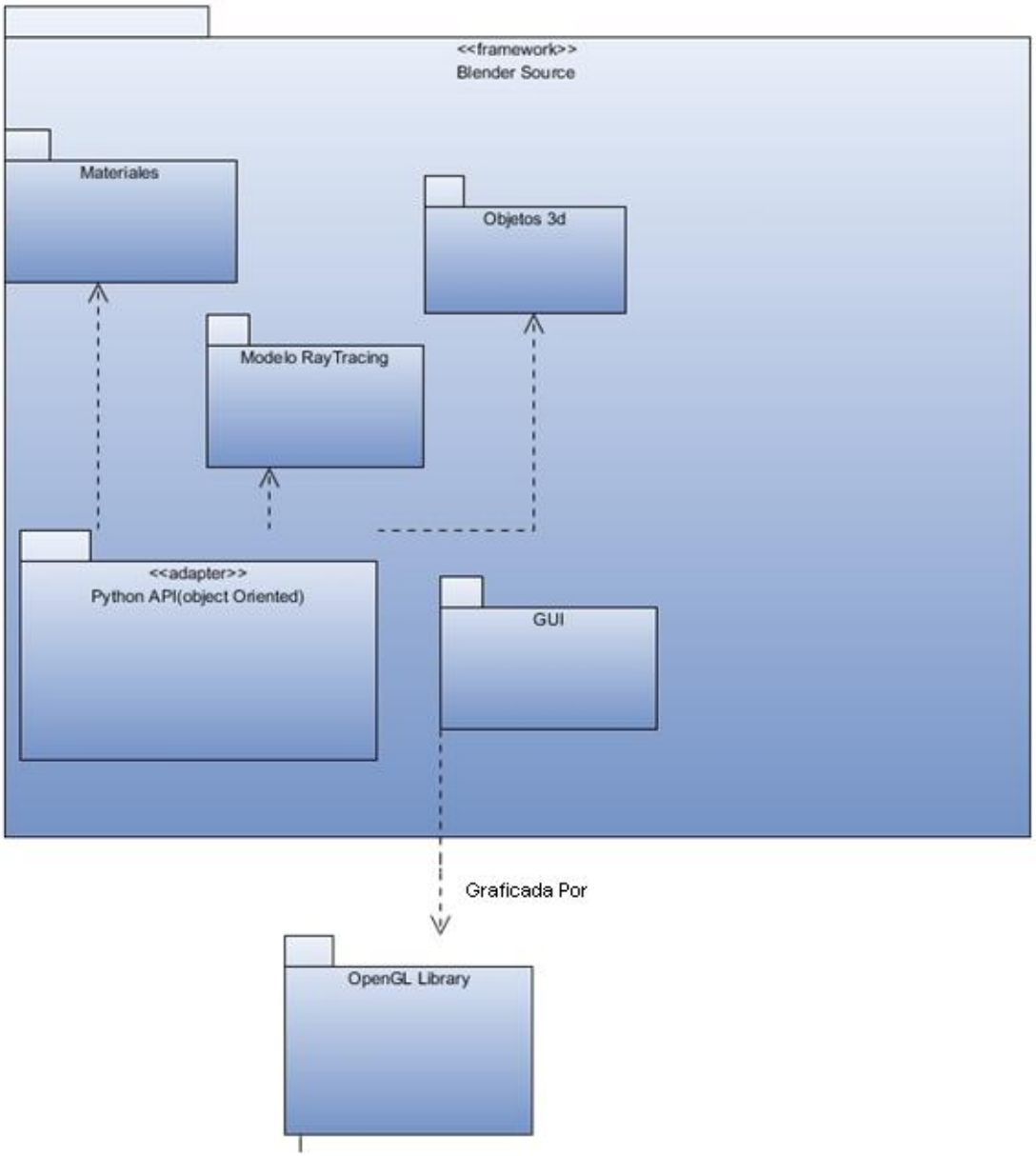
3.3.1 Arquitectura del sistema

"*La arquitectura de software, tiene que ver con el diseño y la implementación de estructuras de software de alto nivel. Es el resultado de ensamblar un cierto número de elementos arquitectónicos de forma adecuada para satisfacer la mayor funcionalidad y requerimientos de desempeño de un sistema*"

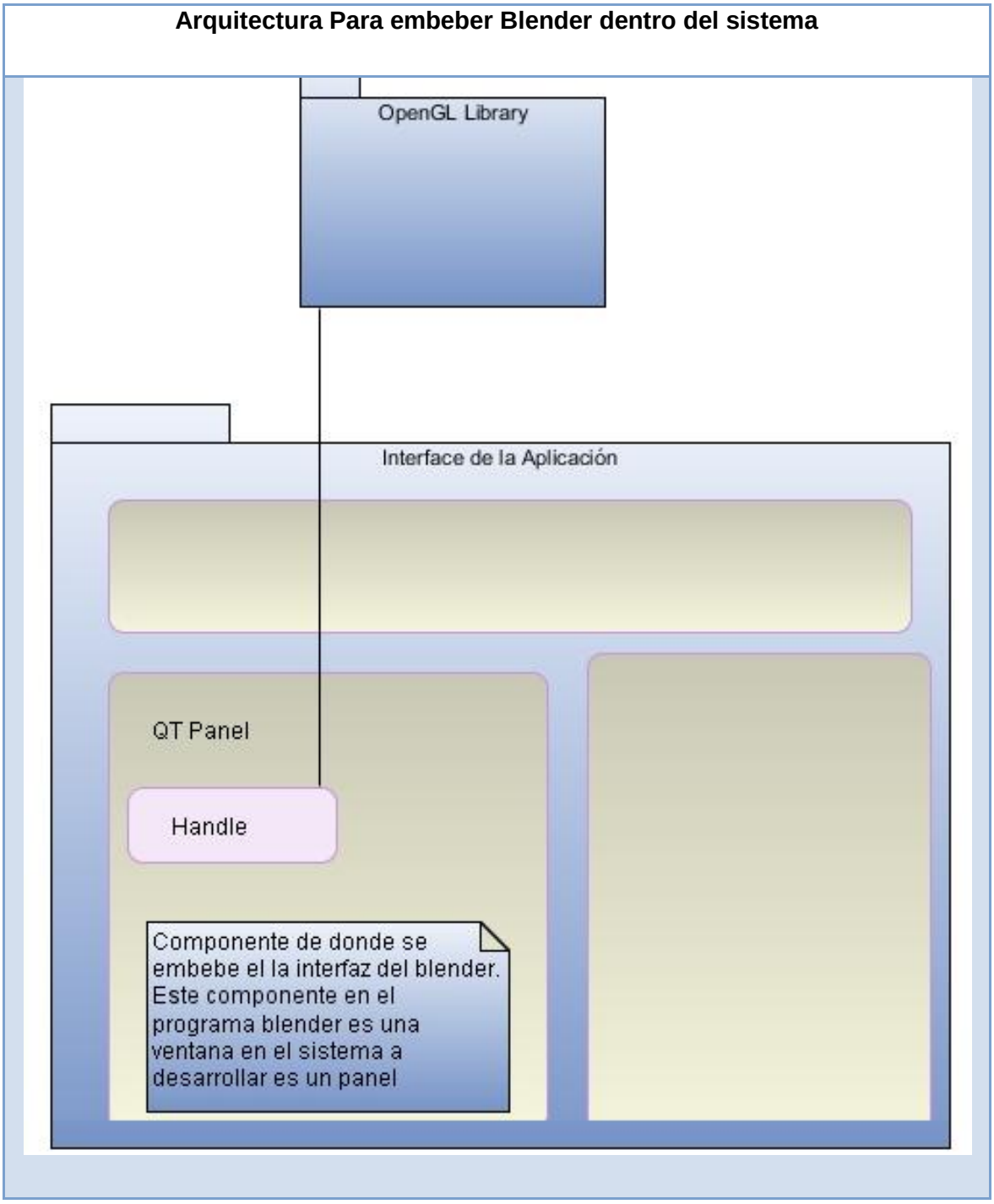
Kruchten, Philippe

La Arquitectura del Software es el diseño de más alto nivel de la estructura de un sistema. Establece los fundamentos para que analistas, diseñadores, programadores, etc. trabajen en una línea común que permita alcanzar los objetivos del sistema de información, cubriendo todas las necesidades. Toda arquitectura de software debe describir diversos aspectos del software. Generalmente, cada uno de estos aspectos se describe de una manera más comprensible si se utilizan distintos modelos o vistas. Es importante destacar que cada uno de ellos constituye una descripción parcial de una misma arquitectura y es deseable que exista cierto solapamiento entre ellos. Esto es así porque todas las vistas deben ser coherentes entre sí, evidente dado que describen la misma cosa. En la figura inferior se describe como se embebe dentro de nuestra aplicación el programa Blender. Básicamente se trata de modificar en el paquete del código fuente del Blender (que debido a su utilidad sirve de framework en nuestro sistema) en módulo de interfaz gráfica que usa la librería OpenGL el punto hacia donde este representa los gráficos redirigiéndolos hacia un panel del sistema.

Arquitectura Para embeber Blender dentro del sistema

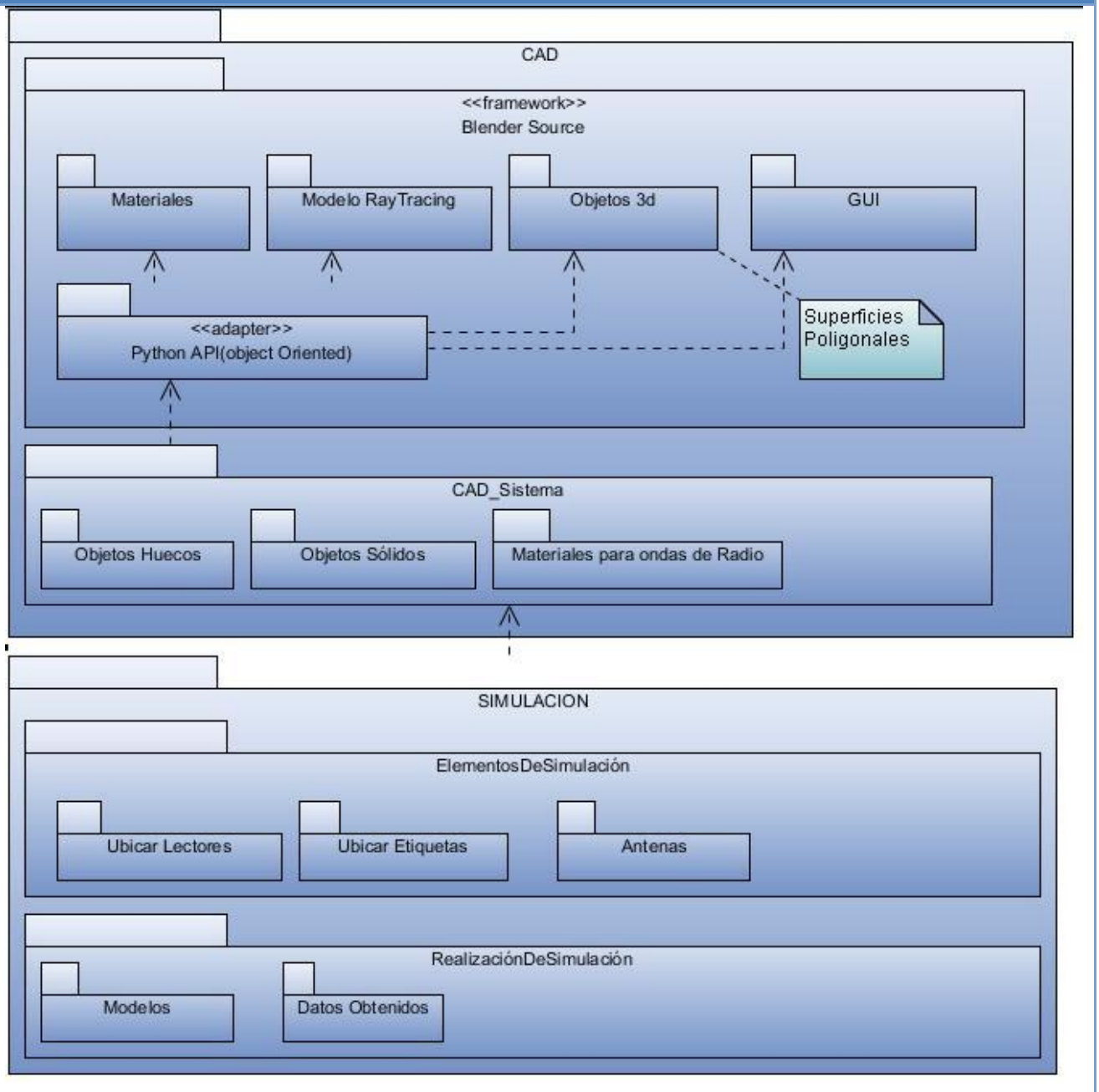


Arquitectura Para embeber Blender dentro del sistema



En la figura inferior tenemos otra vista de la arquitectura del sistema donde se muestran los diferentes módulos que lo componen. Está compuesto por dos módulos principales el módulo de diseño CAD y el módulo de Simulación. El módulo CAD está dividida a su vez en dos submódulos, el submódulo donde se encuentra el código fuente reutilizado del Blender (ajustado para que los requisitos de la aplicación) que se comunica con el submódulo CAD del sistema que a través del API de Python del Blender, el submódulo CAD del sistema realiza los ajustes que no implementa el Blender como definir qué tipo de objeto es (sólido o cuerpo hueco). El de módulo de Simulación está a su vez dividido en dos, el submódulo donde se ubican los elementos de la simulación (Lectores y Etiquetas RFID con sus componentes) y el submódulo donde se realiza la simulación y se obtienen resultados. Para el usuario lograr los resultados que desea de la simulación debe trabajar en las interfaces que provee cada módulo de manera descendente para controlar sus objetos, diseñando primero en la interfaz grafica del Blender embebido realizando los ajustes con las interfaces que provee el módulo CAD del sistema luego ubicando las etiquetas y lectores con sus componentes(antenas y patrones de radiación) implementados en el primer submódulo de Simulación para luego realizar la simulación (escogiendo los modelos a aplicar) en el ultimo módulo de simulación.

Módulos del sistema



3.3.2 Diagrama de Clases del Diseño

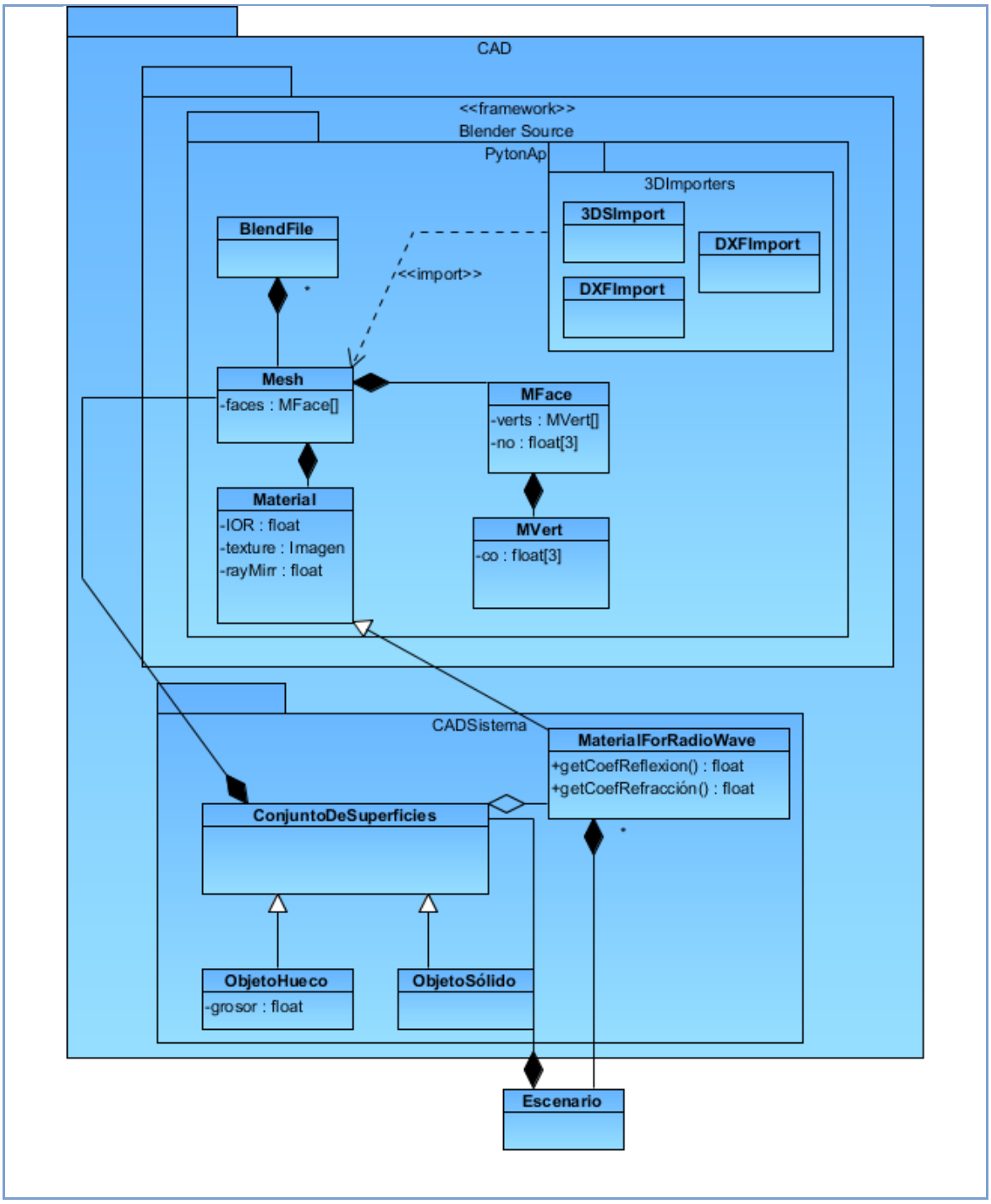
Un diagrama de clases representa las clases que serán utilizadas dentro del sistema y las relaciones que existen entre ellas.

Los elementos básicos que podemos encontrar en un Diagrama de Clases son clases y relaciones entre estas. Existen otros que también pueden mostrarse como paquetes. Los diagramas de clases del diseño expresan para el sistema computarizado la definición de clases como componentes del software.

Normalmente contiene clases, asociaciones y atributos; interfaces con sus operaciones y constantes, además de métodos, información sobre los tipos de los atributos, navegabilidad y dependencias. Los diagramas de clases del diseño expresan para el sistema computarizado la definición de clases como componentes del software.

A continuación mostramos los diagramas de Clases del Diseño correspondientes al sistema:

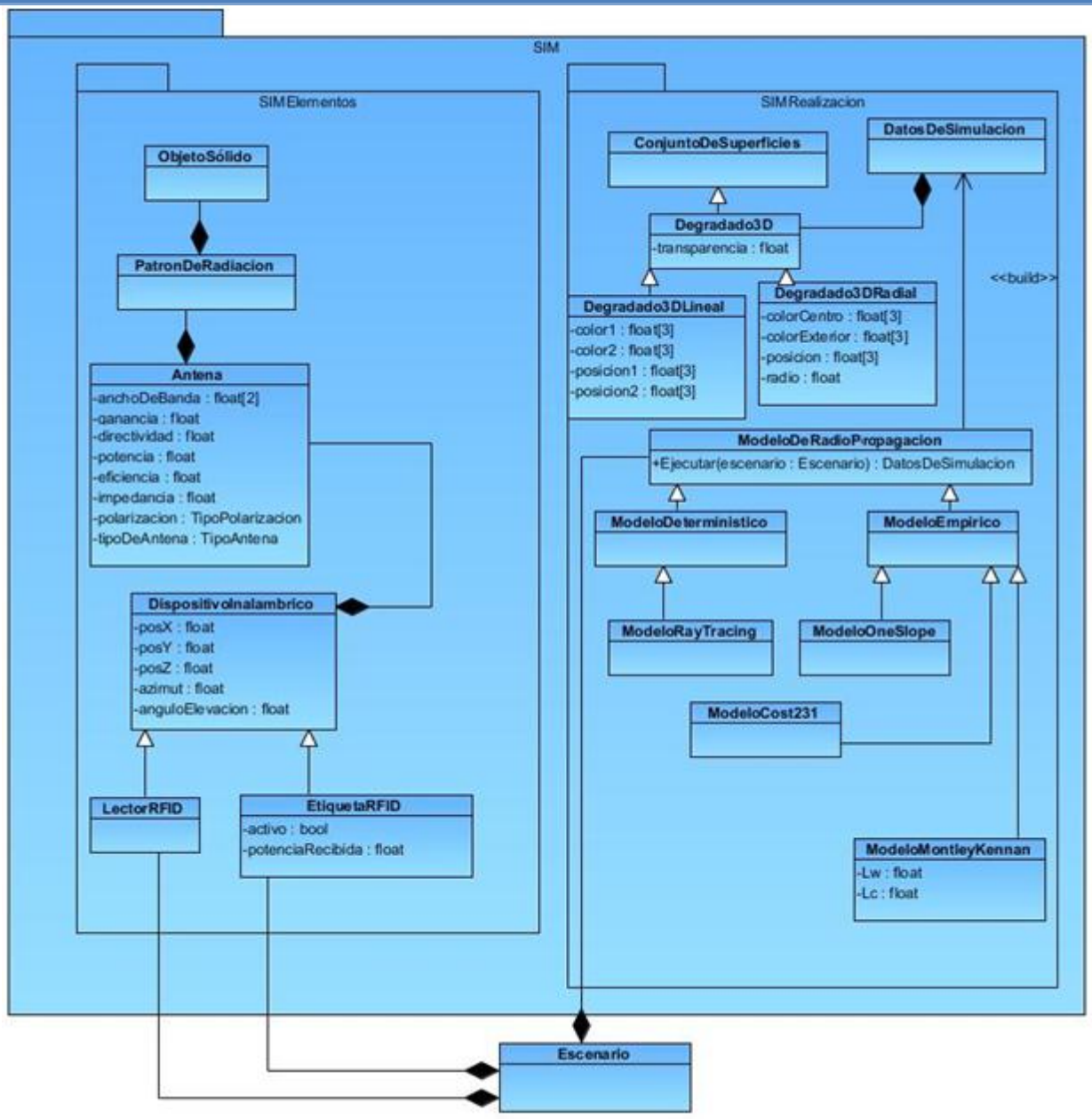
Diagrama de Clases del Diseño. Entidades del módulo CAD


CAD
<<framework>>
Blender Source
PytonAp
3DImporters
3DSImport
DXFImport
DXFImport
BlendFile
<<import>>
Mesh
-faces : MFace[]
MFace
-verts : MVert[]
-no : float[3]
Material
-IOR : float
-texture : Imagen
-rayMirr : float
MVert
-co : float[3]
CADSistema
MaterialForRadioWave
+getCoefReflexion() : float
+getCoefRefracción() : float
ConjuntoDeSuperficies
ObjetoHueco
-grosor : float
ObjetoSólido
Escenario

Descripción de las clases del diseño (submódulo CADSistema)	
ConjuntoDeSuperficies	Clase que representa una agrupación de superficies que definen un objeto en el escenario. Compuestas por objetos del tipo Mesh.
	Atributos: • *meshes:Mesh[]:*Lista de objetos Mesh Métodos: • *getMeshes():Mesh[]* : Obtiene una referencia a la lista de objetos de tipo Mesh que esta posee.
ObjetoSólido	Clase hija de la clase **ConjuntoDeSuperficies** que representa que este define y está validado como el cuerpo sólido en el espacio de tres dimensiones que encierra dicho **ConjuntoDeSuperficies.**
ObjetoHueco	Clase hija de la clase **ConjuntoDeSuperficies** que representa que dichas superficies definen un objeto aplicándole un grosor a estas.
	Atributos: • *grosor: float* : grosor del objeto hueco Métodos: • *setGrosor(grosor:float)*: Establece el grosor • *getGrosor():float* :Obtiene el grosor
MaterialForRadioWave	Clase que hereda de **Material** del framework de Blender, define como se comporta un material con respecto a las ondas electromagnéticas.

<table>
<tr><td></td><td>Métodos:
getCoeficienteDeRefleccion():float : Obtine el coeficiente de reflección del material
getCoeficienteDeRefraccion():float : Obtine el coeficiente de refracción del material
setCoeficienteDeRefleccion(coef:float): Establece el coeficiente de reflexión del material
setCoeficienteDeRefraccion(coef:float) : Establece el coeficiente de refracción del material</td></tr>
<tr><td rowspan="2">Escenario</td><td>Clase que contiene los componentes del escenario en donde se realiza la simulación. Se serializa con fin de guardar el trabajo en disco duro.</td></tr>
<tr><td>Atributos:
ListaDeConjDeSuperfices:ConjuntoDeSuperficies[]
ListaDeMateriales[]:MaterialesForRadioWave
ListaDeLectoresRFID[]:LectoresRFID
ListaDeEtiquetasRFID[]:EtiquetasRFID
ListaDeAntenas:Antenas[]
ListaDePatrones:PatronDeRadiacion[]
ListaDeModelosActivos:ModeloDePropagacion[]
Métodos:
Serializar():FlujoDeBytes: Serializa la información del Escenario para que esta sea guardada en disco.
Se debe implementar método de acceso y modificación (set & get) para todos sus atributos.</td></tr>
</table>

Diagrama de Clases del Diseño. Entidades del módulo SIM



Descripción de las clases del diseño (módulo SIM)	
DispositivoInalambrico	Clase que representa a un dispositivo Inalámbrico genérico
	Atributos • *posX:float* :Posición en el eje X del espacio tridimensional • *posY:float* : Posición en el eje Y del espacio tridimensional • *posZ:float* : Posición en el eje Z del espacio tridimensional • *azimut:float :*Azimut de la antena del dispositivo • *anguloElevacion:float*: Angulo de elevación de la antena del dispositivo. Métodos • Se debe implementar método de acceso y modificación (set & get) para todos sus atributos.
LectorRFID	Clase que representa el dispositivo Lector de RFID
EtiquetaRFID	Clase que representa el dispositivo Etiqueta de RFID

<table>
<tr><td></td><td>Atributos

• activo:bool: Determina si la etiqueta es una etiqueta activa en caso de ser verdadero de otro modo es pasiva.
• potenciaRecibida:float: potencia recibida por la etiqueta calculada una vez efectuada la simulación.

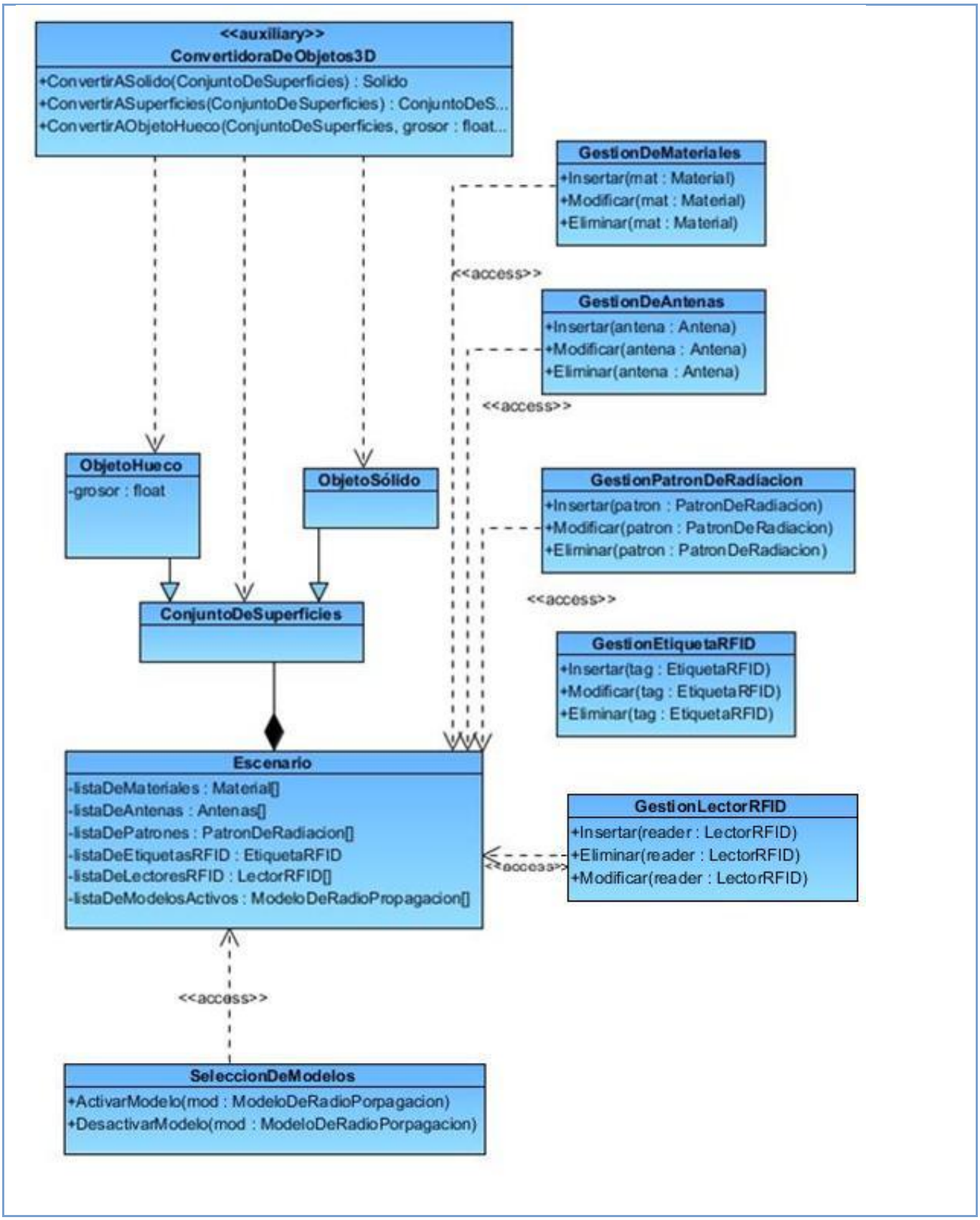
Métodos

• Se debe implementar método de acceso y modificación (set & get) para todos sus atributos.</td></tr>
<tr><td rowspan="2">Antena</td><td>Clase que representa el dispositivo Antena encargado de emitir y recibir ondas electromagnéticas.</td></tr>
<tr><td>Atributos

• anchoDeBanda : float[2]
• ganancia : float
• directividad : float
• potencia : float
• eficiencia : float
• impedancia : float
• polarizacion:TipoPolarizacion:
TipoPolarizacion es un enum {polarización circular mano derecha, polarización circular mano izquierda , polarización vertical , polarización horizontal }
• tipoDeAntena:TipoAntena:
TipoAntena es un enum {direccional ,</td></tr>
</table>

	omnidireccional} • *patron:PatronDeRadiacion* Métodos • Se debe implementar método de acceso y modificación (set & get) para todos sus atributos.
PatronDeRadiacion	Clase que contiene el patrón de radiación radiado por una antena
	Atributos • *lobulos:ObjetoSólido*
ModeloDeRadioPropagacion	Clase abstracta de la cual las clases hijas deben implementar un modelo de radio propagación para realizar la simulación en el escenario.
	Métodos • *Ejecutar(escenario:Escenario):DatosDeSimulacion:* Aplica el modelo de propagación en el escenario y devuelve los resultados de la simulación, le asigna a cada EtiquetaRFID la potencia recibida en esta.
DatosDeSimulacion	Clase que representa los resultados de la simulación a mostrar contiene degradados de colores que representan la potencia recibida en las diferentes zonas del escenario.

Diagrama de Clases del Diseño. Controladoras y elementos asociados


<<auxiliary>>
ConvertidoraDeObjetos3D
+ConvertirASolido(ConjuntoDeSuperficies) : Solido
+ConvertirASuperficies(ConjuntoDeSuperficies) : ConjuntoDeS...
+ConvertirAObjetoHueco(ConjuntoDeSuperficies, grosor : float...
GestionDeMateriales
+Insertar(mat : Material)
+Modificar(mat : Material)
+Eliminar(mat : Material)
<<access>>
GestionDeAntenas
+Insertar(antena : Antena)
+Modificar(antena : Antena)
+Eliminar(antena : Antena)
<<access>>
ObjetoHueco
-grosor : float
ObjetoSólido
GestionPatronDeRadiacion
+Insertar(patron : PatronDeRadiacion)
+Modificar(patron : PatronDeRadiacion)
+Eliminar(patron : PatronDeRadiacion)
<<access>>
ConjuntoDeSuperficies
GestionEtiquetaRFID
+Insertar(tag : EtiquetaRFID)
+Modificar(tag : EtiquetaRFID)
+Eliminar(tag : EtiquetaRFID)
Escenario
-listaDeMateriales : Material[]
-listaDeAntenas : Antenas[]
-listaDePatrones : PatronDeRadiacion[]
-listaDeEtiquetasRFID : EtiquetaRFID
-listaDeLectoresRFID : LectorRFID[]
-listaDeModelosActivos : ModeloDeRadioPropagacion[]
GestionLectorRFID
+Insertar(reader : LectorRFID)
+Eliminar(reader : LectorRFID)
+Modificar(reader : LectorRFID)
<<access>>
<<access>>
SeleccionDeModelos
+ActivarModelo(mod : ModeloDeRadioPorpagacion)
+DesactivarModelo(mod : ModeloDeRadioPorpagacion)

Descripción de las clases del diseño (Controladoras y Auxiliares)	
ConvertidoraDeObjetos3D	Clase auxiliar que convierte un objeto tridimensional en otro.
	Métodos • *ConvertirASolido(ConjuntoDeSuperficies) : Solido* • *ConvertirASuperficies(ConjuntoDeSuperficies): ConjuntoDeSuperficies* • *ConvertirAObjetoHueco(ConjuntoDeSuperficies, grosor : float) : ObjetoHueco*
GestionDeMateriales	Clase controladora encargada de gestionar los materiales del escenario.
	Métodos • *Insertar(mat : Material)* • *Modificar(mat : Material)* • *Eliminar(mat : Material)*
GestionDeAntenas	Clase controladora encargada de gestionar las antenas del escenario.
	Métodos • *Insertar(antena : Antena)* • *Modificar(antena : Antena)* • *Eliminar(antena : Antena)*
GestionPatronDeRadiacion	Clase controladora encargada de gestionar los patrones de radiación del escenario.
	Métodos • *Insertar(patron: PatronDeRadiacion)* • *Modificar(patron: PatronDeRadiacion)* • *Eliminar(patron: PatronDeRadiacion)*

GestionEtiquetaRFID	Clase controladora encargada de gestionar las Etiquetas RFID del escenario.
	Métodos • *Insertar(tag : EtiquetaRFID)* • *Insertar(tag : EtiquetaRFID)* • *Eliminar(tag : EtiquetaRFID)*
GestionLectorRFID	Clase controladora encargada de gestionar los Lectores RFID del escenario.
	Métodos • *Insertar(reader : LectorRFID)* • *Eliminar(reader: LectorRFID)* • *Modificar(reader: LectorRFID)*
SeleccionDeModelos	Clase que controla que modelos de propagación serán aplicados al escenario.
	Métodos • *ActivarModelo(mod: ModeloDeRadioPorpagacion)* • *DesactivarModelo(mod: ModeloDeRadioPorpagacion)*

Diagrama de Clases del Diseño. Clases de la interfaz de usuario

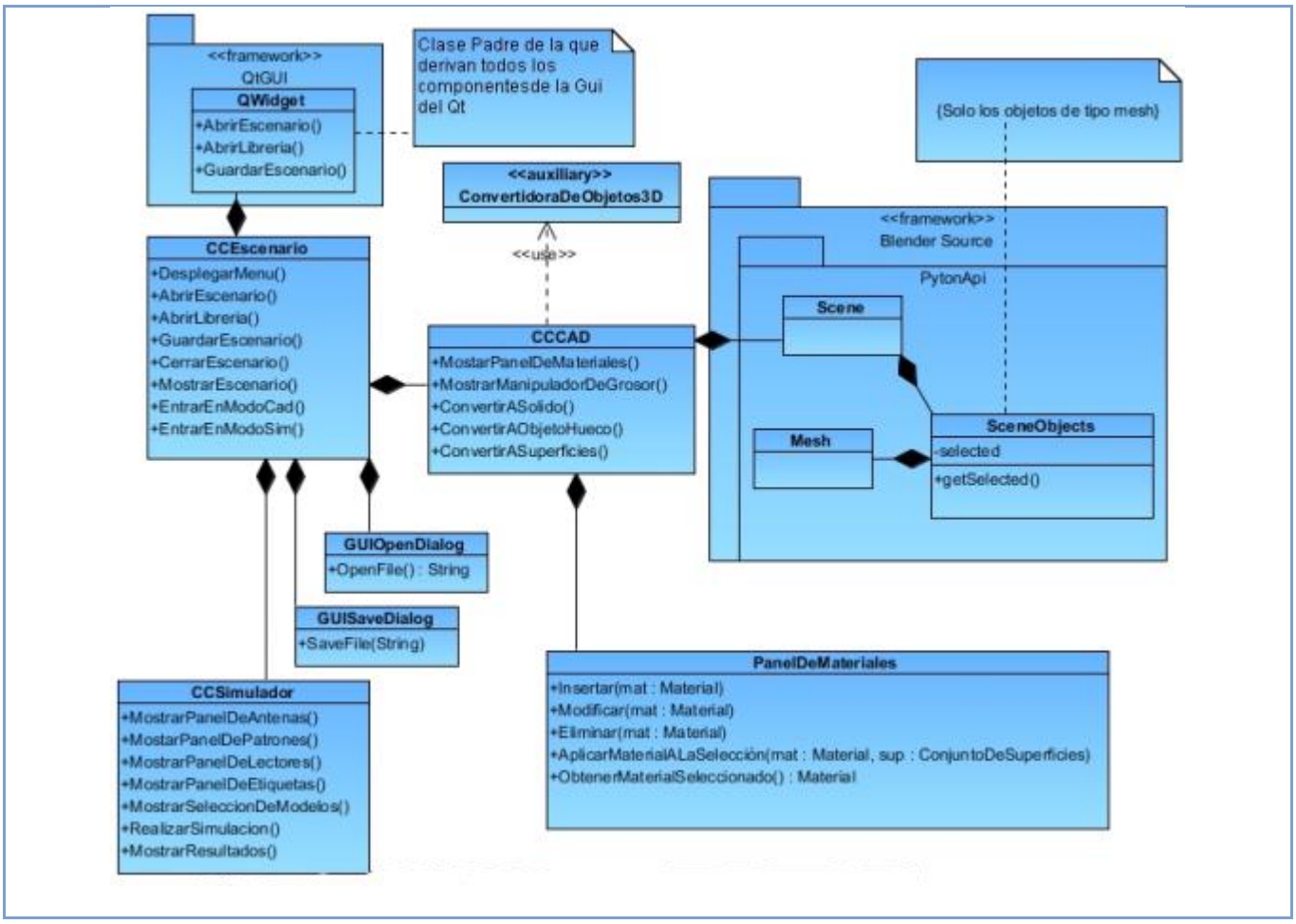
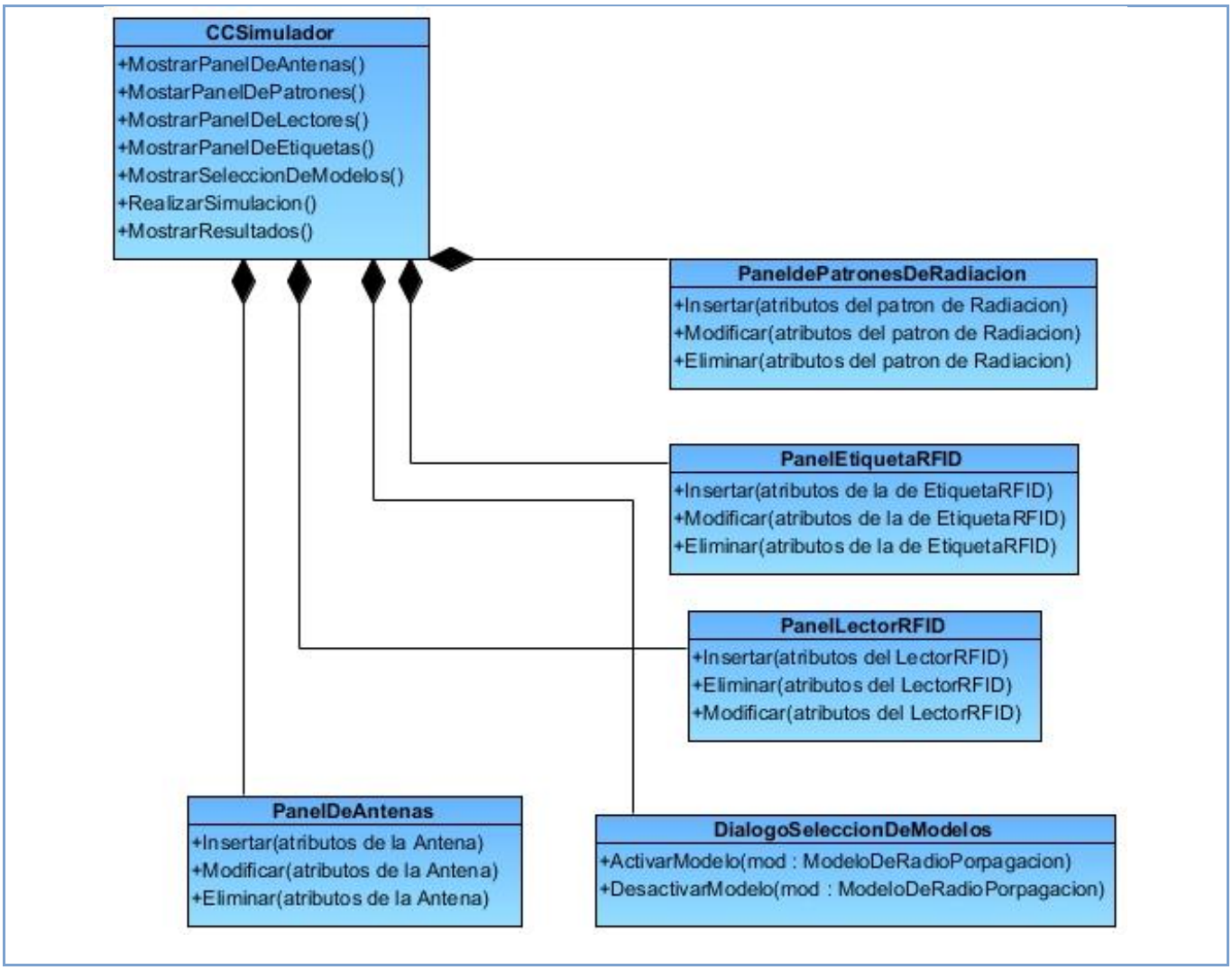
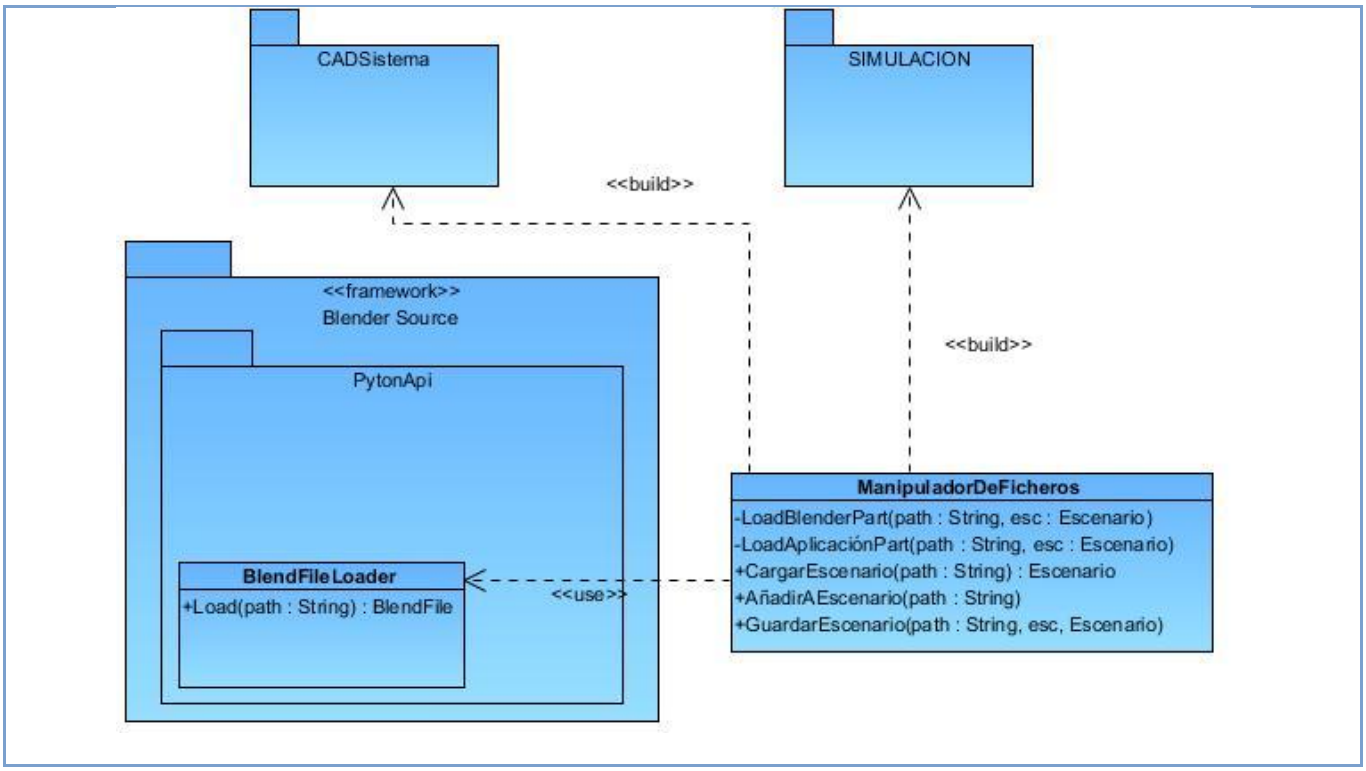


Diagrama de Clases del Diseño. Clases de la interfaz de usuario



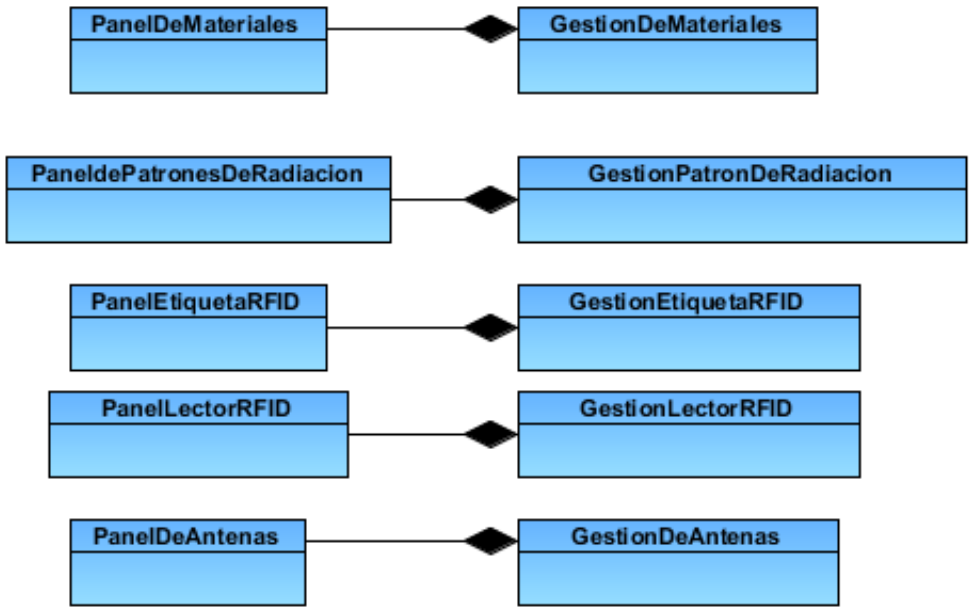
Las clases de la interfaz de usuario controlan los diferentes componentes de la GUI que deben ser mostrados, ocultados y refrescar su información y envían la petición de realizar una operación a las clases controladoras.

Diagrama de Clases del Diseño. Cargador de Ficheros



Descripción de la clase ManipuladorDeFicheros
Manipulador de ficheros es una clase auxiliar encargada de cargar y guardar en disco un escenario de prueba.
Métodos: • *LoadBlenderPart(path : String, esc : Escenario):* Carga del fichero los objetos específicos del framework Blender. • *LoadAplicaciónPart(path : String, esc : Escenario) :* Carga del fichero los objetos específicos de la aplicación. • *CargarEscenario(path:String:BlendFile) : Escenario:* Carga un el escenario a partir de un fichero. • *AñadirAEscenario(path:String:BlendFile) :* Añade los elementos de los catálogos de un escenarion externo al escenario actual. • *GuardarEscenario(path : String, esc, Escenario)* :Guarda el escenario actual en un fichero en disco.

Diagrama de Clases del Diseño. Acoplamiento entre . Clases de la interfaz y gestores



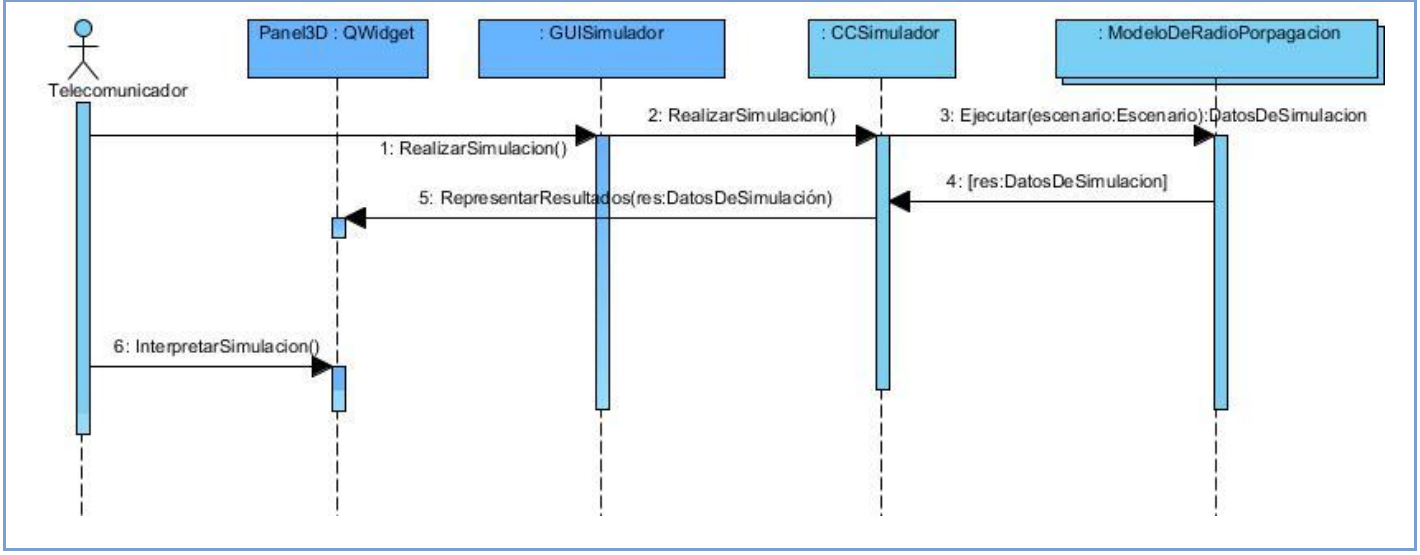
3.3.3 Diagrama de Secuencia del Diseño

Un uso de un diagrama de secuencia es mostrar la secuencia del comportamiento de un caso del uso. Cuando está implementado el comportamiento, cada mensaje en un diagrama de secuencia corresponde a una operación en una clase, a un evento disparador, o a una transición en una máquina de estados

Los diagramas de secuencia en el diseño se realizan a una mayor profundidad, se lleva a cabo un mejor entendimiento del sistema y se detalla este comportamiento mediante objetos y mensajes enviados entre objetos. Lo que permite visualizar cual es el camino que sigue el comportamiento de los Casos de Uso del sistema.

A continuación se muestra el diagrama de secuencia del diseño del caso de uso crítico **Realizar Simulación** correspondiente al sistema:

Diagrama de Secuencia del CU Realizar Simulación



3.4 Conclusiones

El Análisis y Diseño contribuye a una arquitectura estable y sólida, un diseño robusto de la base de datos y a crear un plano del modelo de implementación.

La peculiaridad del diseño es modelar el sistema, encontrar su forma para que soporte todos sus requisitos. Una entrada esencial en el diseño es el resultado del análisis. El modelo de análisis proporciona una comprensión detallada de los requisitos e impone una estructura del sistema que debe conservarse lo más fielmente posible cuando se da forma al sistema.

En este capítulo ha quedado diseñado el sistema en términos de clases del diseño. Se generaron cada uno de los artefactos y diagramas referentes al flujo de trabajo de análisis y diseño. Al concluir el capítulo quedan sentadas las bases para la futura implementación y prueba del sistema.

Conclusiones

En el presente trabajo de diploma, se realizó un estudio de toda la teórica necesaria para comprender uno de los campos más importantes de las Telecomunicaciones, específicamente la propagación de ondas electromagnéticas en el canal de radio. Hoy, esta temática presenta un desarrollo aún incipiente en soluciones software que simulen el fenómeno de la radio-propagación en interiores. Todo esto conllevó a la necesidad de incursionar en esta área, que al ser ajena a la Informática, se debe describir de manera sencilla para propiciar la comprensión por parte de los desarrolladores de software. En la misma, se realizó un estudio de las herramientas, metodologías, tecnologías, notación de modelado y lenguajes de programación, que se ajustan a las características necesarias para llevar a cabo la implementación del software. Luego se concibieron los flujos de trabajo, siguiendo la metodología RUP, de modelado empresarial y requerimientos, los cuales arrojaron los casos de uso del sistema que se validaron a través de los prototipos de interfaz de usuario. Finalmente, se realizó el flujo de análisis y diseño, donde se describe el sistema mediante el diagrama de clases abriendo el paso a una futura implementación. La concepción de este software facilitará el trabajo de los especialistas en Telecomunicaciones, permitiendo obtener mejores resultados en la planificación de radioenlaces RFID. Con esta aplicación se podrá obtener un estimado de los costos de implementación de un sistema RFID, de manera que no suponga un gasto adicional en la planificación del sistema. Además, podrá ser usado de forma académica por los estudiantes de la especialidad de Telecomunicaciones.

Recomendaciones

Al finalizar el proyecto de fin de carrera quedan algunas recomendaciones que pueden servir de punto de partida para mejorar el trabajo:

- Añadir al diseño una arquitectura de plugins que permita ampliar el programa sin recompilarlo de forma que:
  - Se le puedan añadir nuevos modelos de radio-propagación, como los basados en datos estadísticos o implementaciones más eficientes de los modelos que el software trae.
  - Incorporarle al Software otros sistemas de comunicación como WIFI, Redes Tetra etc…
- Implementar el software basado en el análisis y diseño realizado.
- Diseñar e implementar un software de diseño de antenas que interactué con el software permitiendo importar las antenas diseñadas hacia este.
  - Importar antenas diseñadas de los principales software de diseño de antenas existentes en el mercado.
- Continuar con las investigaciones para añadir nuevas funcionalidades al sistema y obtener mejoras en futuras versiones, logrando adecuarlo cada vez más a las necesidades de los usuarios.

Trabajos citados

1. **Alejandro Cervantes Nájera, Pablo Hernández Reyes, Miriam Santiago Jacobo.***Sistema De Información Y Control De Acceso Basado En Tecnología Rfid.* México : s.n., 2008.

2. awe-communications.com. [En línea] [Citado el: 20 de marzo de 2011.] http://www.awe-communications.com/Products/index.html.

3. **Steven J. Fortune, David M. Gay,Brian W. Kernighan, Orlando Landron, Reinaldo A. Valenzuela.***WISE Design of Indoor Wireless Systems: Practical Computation and Optimization.* s.l. : IEEE Computational Science & Engineering.

4. www.siradel.com. [En línea] [Citado el: 20 de marzo de 2011.] http://www.siradel.com/1/volcano-software-suite.aspx.

5. www.edx.com. [En línea] [Citado el: 20 de marzo de 2010.] http://www.edx.com/products/signalpro.html.

6. **Olivares, José Luis Camargo.***Modelo de cobertura para redes inalámbricas de interiores.* Sevilla : s.n., 2009.

7. www.awe-communications.com. [En línea] [Citado el: 20 de marzo de 2011.] http://www.awe-communications.com/Propagation/Indoor/index.htm.

8. **George Roussos, Springer Verlag.***Networked RFID. Systems, Software and Services .* 2008.

9. **Mark Brown, Sam Patadia, SanjivDua.***RFID + Certification.* 2008.

10. **Himanshu Bhatt, Bill Glover.***RFID. Essentials.* s.l. : O'Reilly, 2006.

11. **Dobkin, Daniel M.***The RF in RFID. Passive UHF RFID in Practice.* 2008.

12. **Sanghera, Dr. Paul.***RFID+ Study Guide and Practice Exam.* s.l. : Syngress, 2007.

13. **Yi Huang, Kevin Boyle.***Antennas From Theory to Practice.* 2008.

14. **JACOBSON, I. y BOOCH, G.***El proceso unificado de desarrollo de Software.* Madrid, España : Adison Wesley, 2000.

15. www.omg.org. [En línea] [Citado el: 14 de diciembre de 2011.] http://www.omg.org/gettingstarted/what_is_uml.htm.

16. www.visual-paradigm.com. [En línea] [Citado el: 14 de diciembre de 2011.] http://www.visual-paradigm.com/aboutus.

17. www.python.org. [En línea] [Citado el: 14 de diciembre de 2010.] http://www.python.org/about.

18. lalmada.mayo.uson.mx. [En línea] [Citado el: 20 de marzo de 2011.] http://lalmada.mayo.uson.mx/cpp/La_historia_del_C++.htm.

19. qt.nokia.com. [En línea] [Citado el: 14 de diciembre de 2010.] http://qt.nokia.com/products/developer-tools/.

20. netbeans.org. [En línea] [Citado el: 15 de diciembre de 2010.] http://netbeans.org/features/index.html.

21. python.org. [En línea] [Citado el: 15 de diciembre de 2010.] http://wiki.python.org/moin/IntegratedDevelopmentEnvironments.

22. qt.nokia.com. [En línea] [Citado el: 15 de diciembre de 2010.] http://doc.qt.nokia.com/4.7-snapshot/designer-manual.html.

23. www.blender.org. [En línea] [Citado el: 15 de diciembre de 2011.] http://www.blender.org.

24. crunkish.com. [En línea] [Citado el: 16 de diciembre de 2010.] http://crunkish.com/top-ten-cad-software/.

25. www.autodesk.com. [En línea] [Citado el: 16 de diciembre de 2010.] http://www.autodesk.com/techpubs/autocad/acad2000/dxf/dxf_format.html.

26. www.martinreddy.net. [En línea] [Citado el: 16 de dicembre de 2010.] http://www.martinreddy.net/gfx/3d/OBJ.spec.

27. www.whisqu.se. [En línea] [Citado el: 16 de diciembre de 2010.] http://www.whisqu.se/per/docs/graphics56.htm.

28. www.opengl.org. [En línea] [Citado el: 16 de diciembre de 2010.] http://www.opengl.org/about/overview/#1.

29. qt.nokia.com. [En línea] [Citado el: 17 de diciembre de 2010.] http://qt.nokia.com/products.